

2024583

by Harri Jones

Submission date: 28-Feb-2023 05:34PM (UTC+0000)

Submission ID: 198166688

File name: 16385_Harri_Jones_2024583_181628_1596929568.pdf (4.62M)

Word count: 5144

Character count: 33756

Course: HND Electrical and Electronic Engineering

Module: SGEE5007 Virtual Instrumentation & Control

Portfolio: NI LabVIEW Magnitude Comparator

Name: Harri Jones

University number: 2024583

Date submitted: 28/02/2023

Contents

Element 1.....	6
1. Introduction	6
2. Research	7
2.1 Digital design methods using VHDL and IP integration.....	7
2.2 NI DAQmx	8
2.3 Digital Magnitude Comparator	9
2.4 Analogue to Digital Converter.....	9
2.5 Load Cell (LCKD-1Kg).....	10
2.6 Instrumentation and signal conditioning.....	11
3. Design Requirements	12
3.1 4-bit adder	12
3.2 4-bit mag comparator	14
3.3 8-Bit mag comparator	16
4. Implementation	17
4.1 4-bit adder and comp combined	17
4.2 8-bit comp PLD design	18
5. Testing and Analysis.....	19
5.1 4-bit adder.....	19
5.2 4-bit mag comp	19
5.3 4-bit combination (Adder+Comparator)	20
5.3 8-bit mag comp	22
6. Evaluation.....	24
Element 2.....	25
1. Introduction	25
2. Load cells and signal conditioning.....	26
2.1 Unity gain of instrument amplifier.....	26
2.2 Instrument amplifier with bridge for load cell sensor.	27
2.3 Testing.	29

3. Prototypes.....	30
3.1 10 and 12-bit ADC prototype examples using LabVIEW.....	30
3.2 8-bit ADC with virtual analogue input.....	32
4. Digital Magnitude Comparator.....	33
4.1 Nexys 4 DDR FPGA.....	33
4.2 Magnitude Comparator setup.....	33
4.3 LabVIEW program.....	34
4.4 Testing.....	35
5. Evaluation.....	37
Element 3.....	38
1. Introduction.....	38
2. Design Requirements.....	38
4. Testing and Analysis.....	40
5. Evaluation.....	41
6. Conclusion and Recommendations.....	42
Bibliography.....	43
6.Appendix.....	45
6.1.1 4-bit code.....	45
6.1.2 4-bit test parameters.....	45
6.1.2 PLD VHDL export.....	46
6.1.3 Test Bench Export.....	47
6.1.4 Mag4bit constraints extract.....	47
6.2.1 – Load Cell testing.....	48
6.3.1.....	54

A comprehensive table of contents

Figure 1 – Intellectual Properties	7
Figure 2 - DAQmx LabVIEW	8
Figure 3 - CD4063 Comparator	9
Figure 4 - ADC	10
Figure 5 - Half Adder Block Diagram	12
Figure 6 - Full Adder Block Diagram	12
Figure 7 - Full Adder Truth Table	12
Figure 8 – 4-bit adder hierarchy	13
Figure 9 – Simulation	13
Figure 10 - IP Integrator design.....	13
Figure 11 - Mag Comparator PLD	14
Figure 12 - Mag4bit comparator.....	14
Figure 13 - Mag4bit IP block	15
Figure 14 - PLD Testing.....	16
Figure 15 - Combi Hierarchy	17
Figure 16 - Combi IP Block	17
Figure 17 – 8-bit PLD design.....	18
Figure 18 - 4-bit simulation test.....	19
Figure 19 - Simulation Tests	19
Figure 20 - 4-bit AETB test.....	20
Figure 21 - 4-bit ALTB test	20
Figure 22 - AGTB.....	21
Figure 23 - 4-bit Test Results	21
Figure 24 - FPGA Less Than	22
Figure 25 - FPGA Equal To.....	22
Figure 26 - FPGA Greater Than.....	23
Figure 27 8-bit test table	23
Figure 28 - Unity Gain	26
Figure 29 - 1KG Load Cell Amplifier.....	28
Figure 30 - Load Cell Graph.....	29
Figure 31 - 10-bit ADC (2.5).....	30
Figure 32 - 10-bit ADC (5).....	30
Figure 33 - 12-bit ADC (2.5).....	31
Figure 34 - 12-bit ADC (5).....	31
Figure 35 - 8-bit ADC	32
Figure 36 8-Bit ADC 2nd test	32
Figure 37 – Pins.....	33

Figure 38 - FPGA connected to NI ELVIS	33
Figure 39 - Mag comb LabVIEW	34
Figure 40 - Greater Than	35
Figure 41 - Equal to	35
Figure 42 - Less Than	36
Figure 43 - Data Logging Block.....	38
Figure 44 - Data Logging Graph.....	40

Extensive list of figures

Element 1

1. Introduction

One of the purposes of element 1 is to gain knowledge from the research which will assist in gaining a better overall understanding of the overall system. VHDL and IP technology alongside the FPGA Nexys 4 DDR board will be used to design, build and test the 4-bit magnitude comparator. An 8-bit magnitude comparator will also be implemented and tested using VHDL and the FPGA Nexys 4 DDR board.

The 4 bit adder should be included in the introduction. You must be aware of the separate tasks required.

2. Research

2.1 Digital design methods using VHDL and IP integration.

VHDL (VHSIC Hardware Description Language) is used to describe the structure and behaviour of digital systems. The software can be used to program the physical circuits on a FPGA (Field Programmable Gate Array) chip. The chips can range from a few gates to many complex integrated circuits. It is widely used in digital systems, for example digital signal processing systems, embedded systems and communication systems. (Nandland, 2018)

VHDL can be used in LabVIEW by using the FPGA module, graphical programming techniques are used to program the FPGA devices.

The FPGA module allows for custom IP cores to be created using VHDL. IP blocks can be used, these can hold a whole system instead of single components (See figure 1).

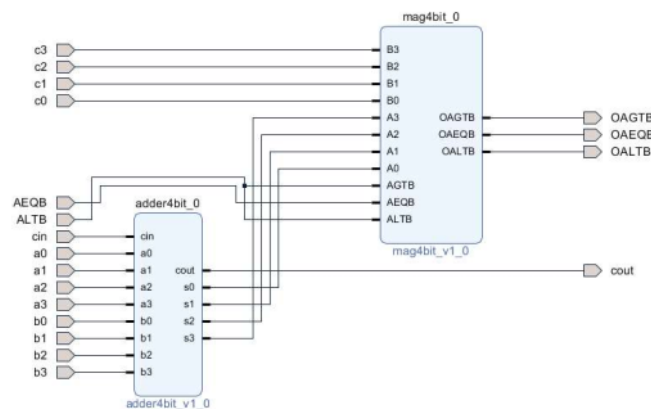


Figure 1 – Intellectual Properties

An IP block is a pre-designed and tested component or block of circuit. It is re-usable and able to be integrated into larger designs. Once the IP block is designed it must be packaged into a format which can be integrated and easily reused into other designs. Once packaged it must be tested using the Vivado simulation and verification tools. As the IP block is reusable this provides advantages such as improved reliability, a low risk of design errors and shorter design cycles. (Semiengineering, 2019)

Good significant research carried out here with relevant sources, and good example of IP system.

2.2 NI DAQmx

NI DAQmx (Data acquisition Measurement extension) is used for controlling and configuring DAQ hardware devices. These devices are used in acquiring and measuring data from instruments and sensors. (MathWorks, 2022)

LabVIEW allows users to create interfaces for their data acquisition applications, this involves tools such as graphical indicators, block diagrams, front panels and controls. LabVIEW also contains analysis and visualization tools for example charts, graphs and data logging. (NationInstruments, 2022)



Figure 2 - DAQmx LabVIEW

Figure 2 shows examples of functions within LabVIEW.

Create Channel – Virtual channels can be implemented, from digital to analogue channels. This function is used to create and configure the data acquisition channels or control applications. Various configuration parameters such as voltage, temperature and units can be specified.

Read – This is used to read the data acquisition hardware device and acquire its data for example an input device or sensor. The function can be specified to read several samples and other parameters.

Write – The output signals from a data acquisition hardware device can be generated using this function. Data is written to the channels configured; the function involves specifying the samples to write and the timeout values.

Clear – This function is used to free up system memory and allows for proper shutdown of the data acquisition hardware device. Data that is buffered or queued will also be cleared.

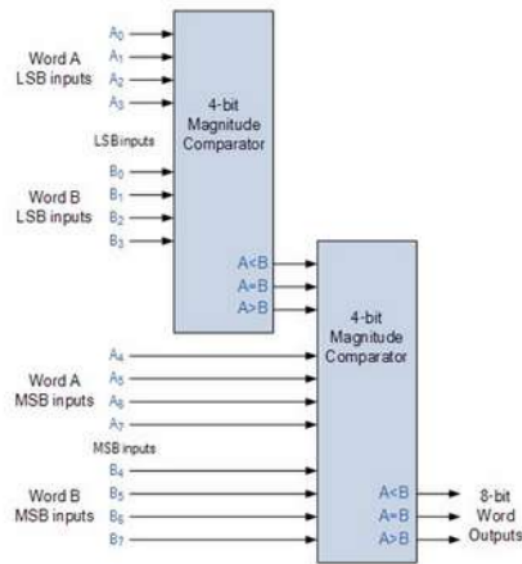
Start – Used to start the data acquisition for the task. The start of the operation is triggered on the configured channels and data is generated or acquired depending on the settings.

(NationInstruments, 2019)

DAQ program blocks have been explained and are significant for the practical in this portfolio. There are aother blocks that should have been include such as formula blocks.

2.3 Digital Magnitude Comparator.

A digital magnitude comparator is a digital circuit which performs a comparison of two binary numbers and determines which one is less than, greater than or equal to. The two binary digits are compared in each position of the two numbers. A basic block can consist of a two input XOR gate. The output is 1 if both inputs are different and 0 if they are the same. Connecting XOR gates in a particular formation will create a digital magnitude comparator. (Pandey, 2022)



Significant points are explained using a relevant supporting source and example

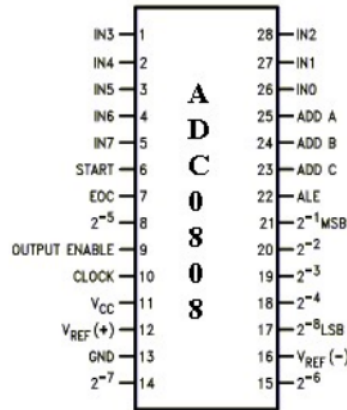
Figure 3 - CD4063 Comparator

Figure 3 shows a cascading CD4063 comparator, most data processing consists of 8 to 16 bits. Cascading multiple units through the cascading inputs is done to compare large BCD or binary numbers. The least significant comparator outputs are connected to the inputs of the more significant comparator. Figure 3 shows the two comparators connected and its purpose is to compare the two 8-bit words. (Microcontrollerslab, 2022)

2.4 Analogue to Digital Converter.

An ADC is an electronic device which converts an analogue signal for example current or voltage into digital signals. These digital signals can be processed by the digital circuit or a computer. It operates by using a continuous analogue signal and samples it at regular intervals which measures the current or voltage level at each interval. The measurements can then be converted into binary digits which represent the original analogue signal.

The resolution and accuracy of the ADC determine how well the digital signal represents the analogue signal. The speed of the ADC can also impact the performance of the overall system.



The ADC methods such as pipeline, flash and SAR should be included

Figure 4 - ADC

Figure 4 shows an ADC0808, it consists of 8 analogue inputs and 8 digital outputs. This ADC can be used to monitor a maximum of 8 transducers by using a single chip. This is useful as it removes the need for external zero and full-scale adjustments. The device is suitable for applications such as automotive and process and machine control as it offers high accuracy, high speed and minimal temperature dependence. (Elprocus, 2022)

2.5 Load Cell (LCKD-1Kg)

A LCKD-1Kg is a type of sensor which measures a weight or force. This one is designed for measuring up to 1Kg with high precision and accuracy. Strain gauges are used which are sensitive elements that change their resistance accordingly to the deformation, to measure the force applied to the load cell. The resistance change can then be converted into a signal which can be read by the data acquisition.

Load cells are widely used in applications involving weighing systems, process control, material testing and force measurement. (Omega, 2023)

This should include a supporting image

2.6 Instrumentation and signal conditioning.

Signal conditioning is the process where the input signal is modified to prepare it for further processing or measurement. The input signal can be a current, voltage or an electrical or physical signal. The modification can consist of a conversion to a different format, amplification or filtering. It is used to improve the quality of the signal and minimize errors in measurement or analysis. The signals can often be noisy, weak or unsuitable for processing tasks.

Signal conditioning techniques:

Filtering – This removes any interference or unwanted noise from the signal by using high-pass, low-pass or band-pass filters.

Analogue to digital conversion – This converts the analogue signal to a digital signal which can then be processed by a digital device or computer.

Amplification – The amplitude or gain of the signal can be increased to make it easier to process or more detectable.

Isolation – The signal is separated from a high current or voltage environment; this prevents any damage or interference.

Linearization – The non-linearities in the signal can be corrected which improves the accuracy and linearity in measurements. (HBM, 2020)

A good understanding demonstrated with relevant source. A supporting image should be included

3. Design Requirements

3.1 4-bit adder

A 4-bit adder is a digital circuit which adds up two 4-bit binary numbers. Performing an arithmetic operation of addition which produces a 4-bit sum as the output.



Figure 5 - Half Adder Block Diagram

Figure 5 shows a half adder block diagram, the purpose of a half adder is to add two single bit numbers. They don't have a scope of adding the carry bit resulting from the addition of previous bits, due to this full adders are used.

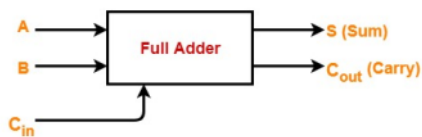


Figure 6 - Full Adder Block Diagram

Inputs			Outputs	
A	B	C _{in}	C _{out} (Carry)	S (Sum)
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 7 - Full Adder Truth Table

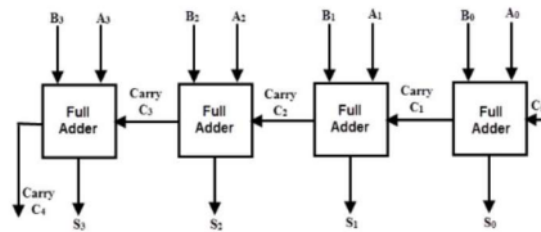
Figure 7 shows a full adder block diagram. This is a combinational logic circuit and its purpose is to add the two single bit numbers with a carry. It contains 3 inputs and 2 outputs.

Input – A, B, C_{in}. These will either be 0 or 1.

Output – S (sum), C_{out} (carry).

The point is made about the building blocks for the 4 bit adder

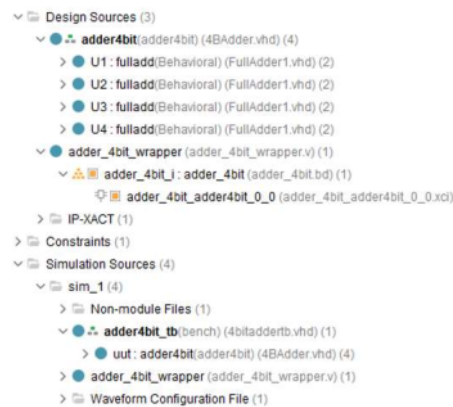
Figure 8 shows a 4-bit adder hierarchy. In this diagram the input Cin is carried from one full adder to the next.



The block diagram is correct and is significant for the VHDL design.

Figure 8 – 4-bit adder hierarchy

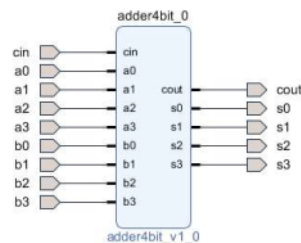
Figure 9 shows the 4-bit adder which is made up of four full adders. Within each adder is 2 half adders.



This hierarchical listing here should be explained

Figure 9 – Simulation

Figure 10 shows the IP integrator design.

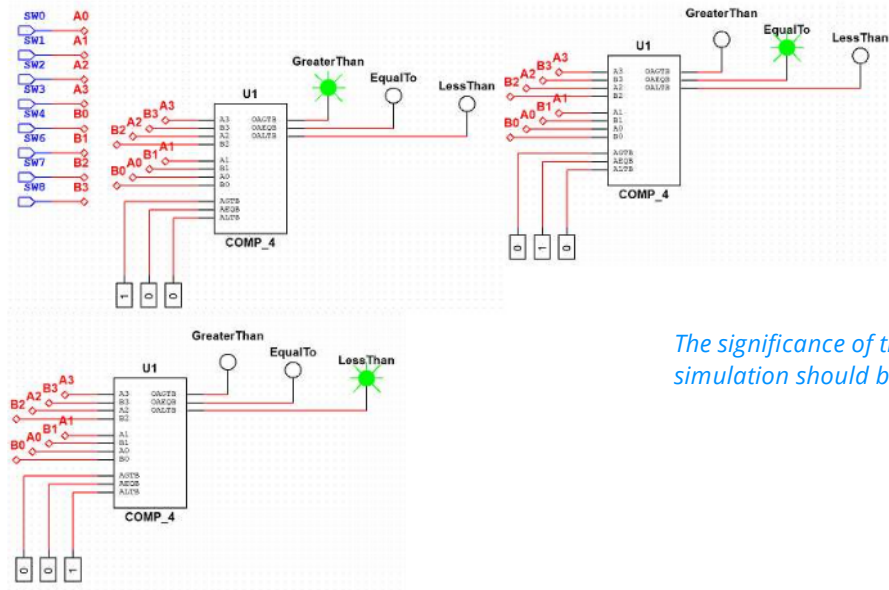


There is no point in these images without an explanation

Figure 10 - IP Integrator design

3.2 4-bit mag comparator

Figure 11 shows the 4-bit mag comparator PLD design. The VHDL was copied from the PLD design and placed in Vivado.



The significance of this 4 bit comparator simulation should be pointed out

Figure 11 - Mag Comparator PLD

Figure 13 shows the mag 4-bit comparator hierarchy in Vivado.

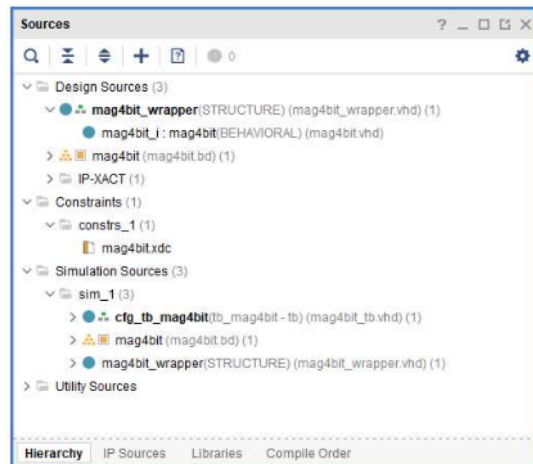
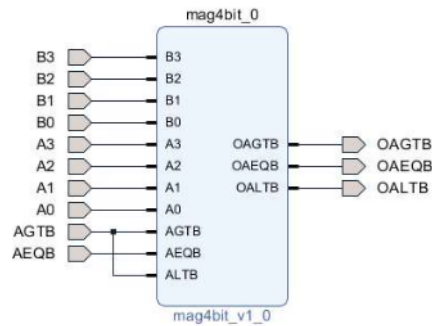


Figure 12 - Mag4bit comparator

Appendix 6.1.3 shows the code placed in to the mag4bit. This code was then placed in a VHDL testbench generator. This could then be simulated to be tested.

An XDC constraints file was created (see figure 13). Appendix 6.1.5 shows an extract from the code.



The IP design is correct, however the explanation is poor

Figure 13 - Mag4bit IP block

The IP block consists of 11 inputs and 3 outputs. The 'Less Than' input is connected to the 'Greater Than' input.

3.3 8-Bit mag comparator

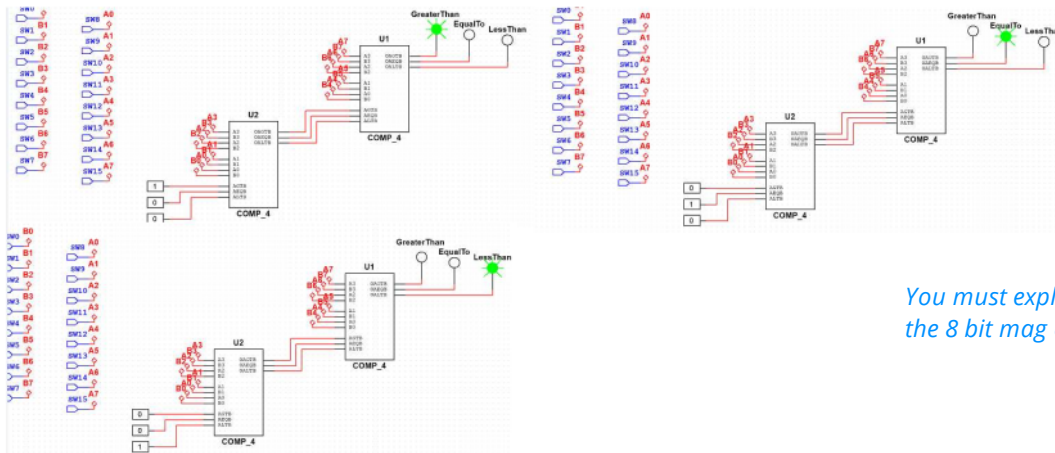


Figure 14 - PLD Testing

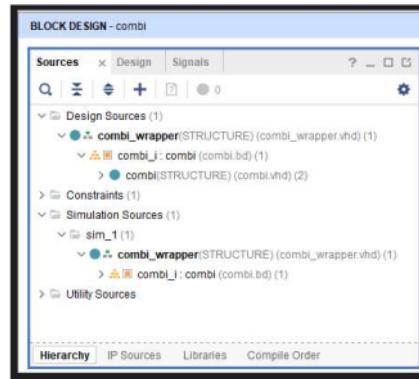
You must explain the purpose for the 8 bit mag comparator

Figure 14 shows the 8-bit comparator PLD design. Probes were added to see the tests visually working.

4. Implementation

4.1 4-bit adder and comp combined.

Figure 15 shows the combined hierarchy.

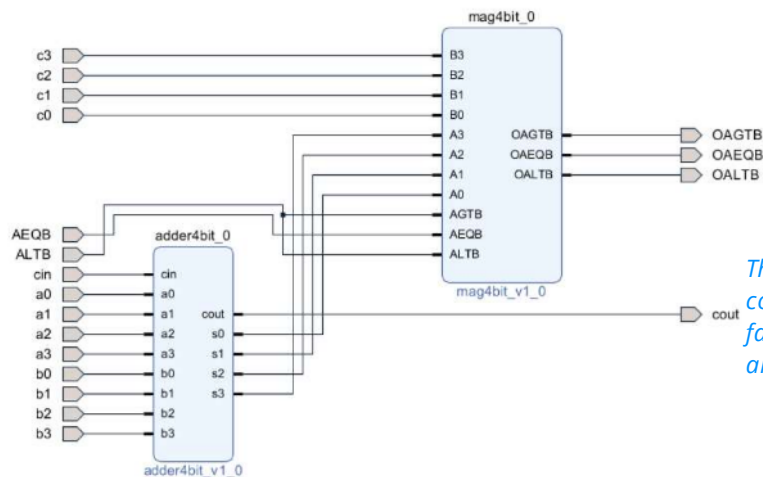


This does not show the hierarchy

Figure 15 - Combi Hierarchy

The IO was created from an .xdc text file. Appendix 6.1.4 for extract of code.

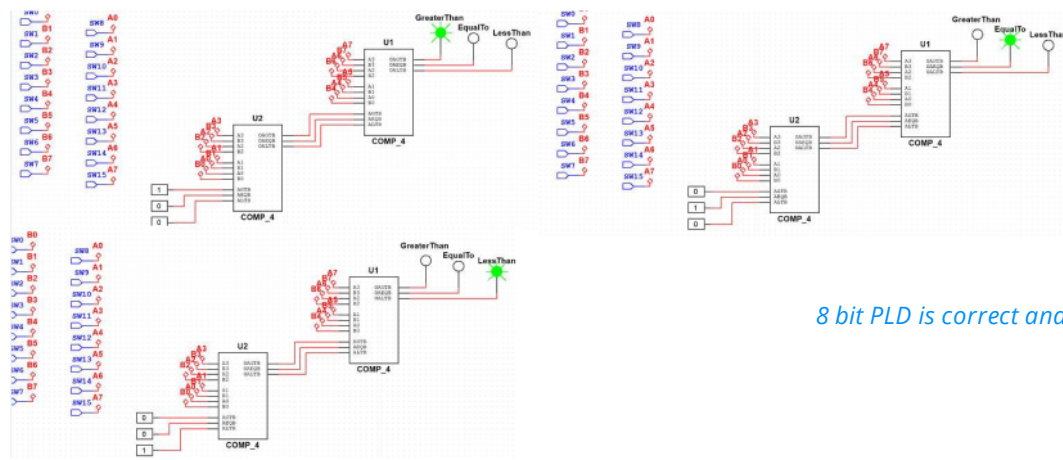
Figure 16 shows the combination of the mag4bit IP block and adder4bit IP block. Connections are made between the outputs of the adder4bit and inputs of the mag4bit.



This combination IP design is correct. However, You have failed to explain the 4 bit adder and its function

Figure 16 - Combi IP Block

4.2 8-bit comp PLD design



8 bit PLD is correct and tested

Figure 17 – 8-bit PLD design

The 8-bit PLD design was then downloaded on to the FPGA board.

There should be an explanation here!

5. Testing and Analysis

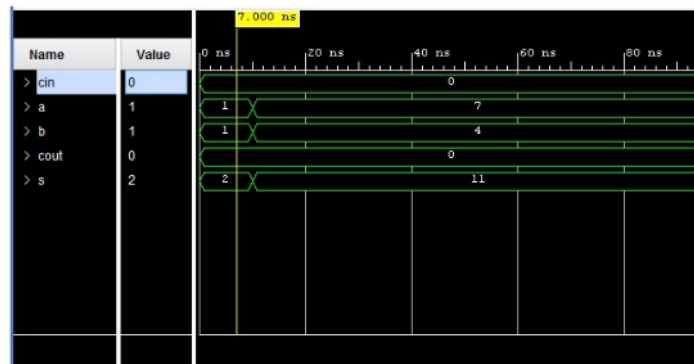
5.1 4-bit adder

A test bench was created from a generator online.

Appendix 6.1.1 shows the code placed in to the 4-bit adder file.

Test parameters were then used to perform an analysis (see appendix 6.1.2).

After implementing the test parameters, the simulation was ran(see figure 18).



Full range is not tested. The carry bit 'cout' is not set

Figure 18 - 4-bit simulation test

If the sum is 16 then the Cout will go to 1. In this case the sum is less than 16 meaning that the Cout logic is 0.

5.2 4-bit mag comp

Appendix 6.1.4 shows the code after placing it in the test bench generator, [Appendix 6.1.3](#)



Full range has been tested

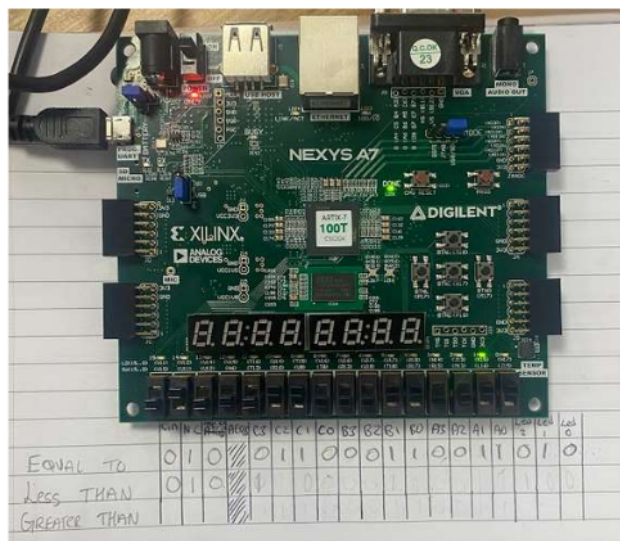
Figure 19 - Simulation Tests

Figure 19 shows the mag 4-bit comparator test. The test shows that $A = 12$ and $B = 4$. This is giving OAGTB a value of 1.

This needs to be explained further and include all 3 output conditions

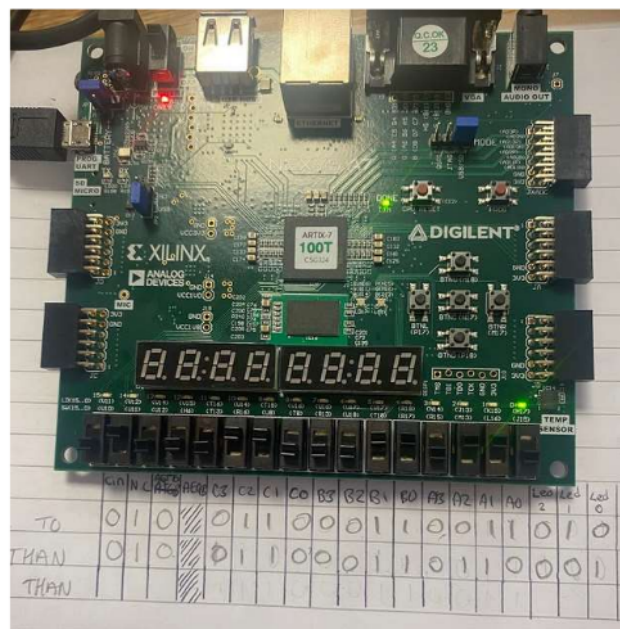
5.3 4-bit combination (Adder+Comparator)

The program was then downloaded on to a Nexys 7 FPGA board for testing.



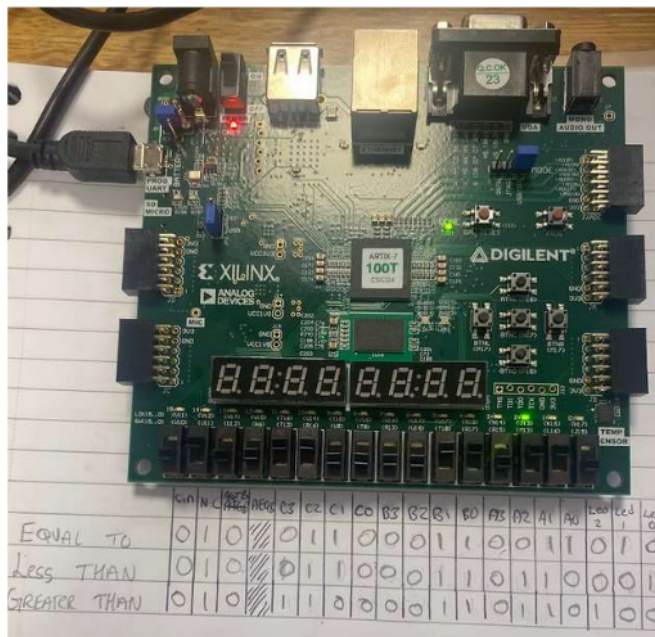
The test is successful and the mapping indicates the io conditions

Figure 20 - 4-bit AETB test



Correct result with io conditions set

Figure 21 - 4-bit ALTB test



Test is successful with io conditions set

Figure 22 - AGTB

The tests were successfully carried out and the results are below (Figure 22)

Good use of table for results

Inputs			B1		B		A		Outputs			
Cin	AGTB	AEQB	Binary	Decimal	Binary	Decimal	Binary	Decimal	ALTb	AEQB	AGTB	Cout
0	0	1	0110	6	0011	3	0011	3	0	1	0	0
0	0	1	0110	6	0011	3	0110	6	1	0	0	0
0	0	1	1100	12	0011	3	0110	6	0	0	1	0

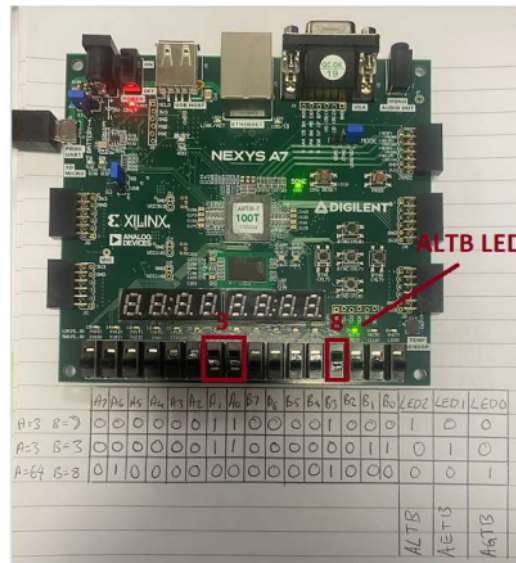
Figure 23 - 4-bit Test Results

Explanations should be included

5.3 8-bit mag comp

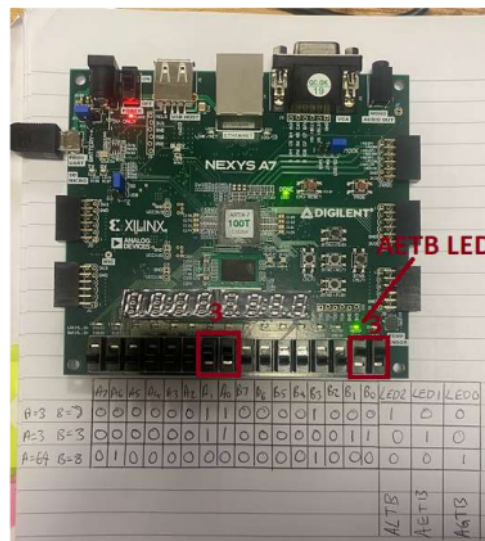
The program was then downloaded on to a Nexys 7 FPGA board for testing.

This explanation is incomplete



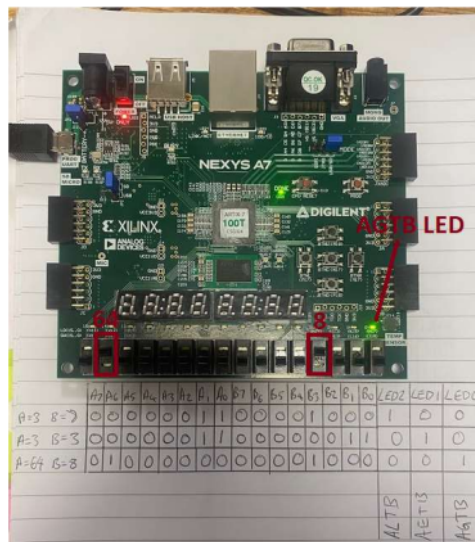
Test is successful and io conditions set

Figure 24 - FPGA Less Than



Test is succesful with io conditions set

Figure 25 - FPGA Equal To



Test is successful with io conditions set

Figure 26 - FPGA Greater Than

The 3 tests carried out were successful. Figure 26 shows a testing table.

Good use of table for results

A		B		Output		
Bit	Numeric	Bit	Numeric	ALTB	AEQB	AGTB
00000011	3	00001000	8	1	0	0
00000011	3	00000011	3	0	1	0
01000000	64	00001000	8	0	0	1

Figure 27 8-bit test table

There should be an explanation of the tests carried out

6. Evaluation

The VHDL prototypes were used to generate a physical result from the IP design systems by pairing them with the FPGA board. The reliability of the FPGA output data was proven from the digital design with 4-bit adder and comparator. A benefit of having the IP design is that it can be generated and implemented for any systems using interfacing.

The FPGA board was used to test the 8-bit comparator and the results prove that the outputs are reliable. The 8-bit comp can be implemented with DAQmx and LabVIEW using the I/O ports on the FPGA and DAQmx. If the FPGA input values are steady, a reliable output will be generated.

Evaluation includes significant points

Element 2

1. Introduction

element

This portfolio is concentrated on LabVIEW (Laboratory Virtual Instrument Engineering Workbench). The system design platform is used widely in industrial automation, engineering and scientific research. LabVIEW uses a graphical language to create applications by placing and connecting components to build the programs functionality. An NI Elvis II was used with the 8-bit magnitude comparator created in portfolio 1.

Loadcells are also within portfolio 2, investigating how the load cell reacts when the weight is altered, the signal was conditioned to gain a weight measurement through the digital signal.

Good introduction that demonstrates an understanding of task requirements

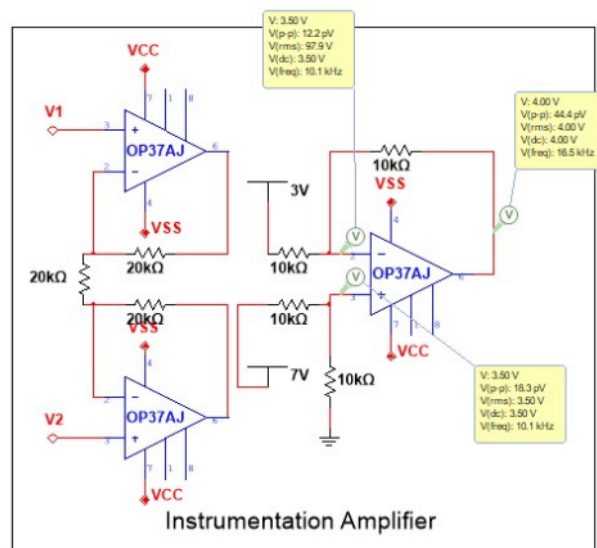
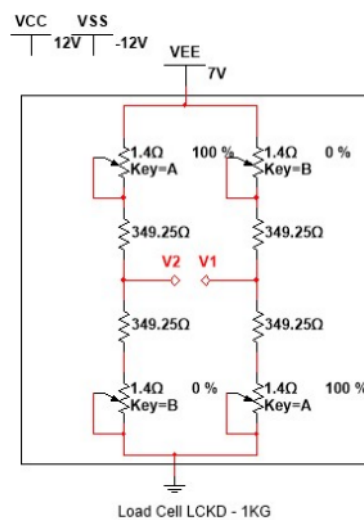
2. Load cells and signal conditioning.

2.1 Unity gain of instrument amplifier

Figure 28

Figure 24 shows the instrumentation amplifier circuit. As seen below the instrument amp is not connected to the amplifier part of the circuit. 3V and 7V voltages were loaded to the inputs, the amplifier output with unity is 4V. The amplifier is a differential amplifier with two inputs +ve and -ve. A unity gain of 4V is achieved due to the difference of both inputs (7V-3V).

Good explanation



Good test for unity gain

Figure 28 - Unity Gain

2.2 Instrument amplifier with bridge for load cell sensor.

The amplifier previously used (Figure 28) will be paired with an Omega LCKD load cell. The load cells are very compact and robust, allowing them to be used in tight spaces where traditional load cells couldn't be used. They come in a range of different capacities from 25lb to 10,000lb, in this case a 1Kg is used. The load cell uses strain gauge technology, giving it a high accuracy (up to 0.03%).

Below are the specifications of the LCKD load cell.

LCKG – 1KG

Excitation: 5Vdc, 7Vdc max

Output: 2mV/V nominal

Load cell spec is identified

Unstressed resistance 350Ω

The load cell will be zero grams (0V) at 50% on all resistors.

The cell will be fully loaded 1000 grams (5v) at 100% and 0% resistor settings in opposite corners.

To check the gain of the amplifier

$$\text{Gain} = (1+2 * (20/10)) * (5000/50) = 5$$

*Good choice of resistor values for instrument amplifier.
The gain calc works out correctly and meets requirements*

$$\text{Buffer Gain} = (1+2 * (20/10))$$

$$\text{Diff Amp Gain} = (5000/50)$$

$$\text{Overall gain} = \text{The Buffer gain} * \text{diff amp gain} = 5*100 = 500$$

Figure 29 shows the 1KG load cell amplifier circuit. The resistors begin at 50%.

This should include that these are the initial conditions

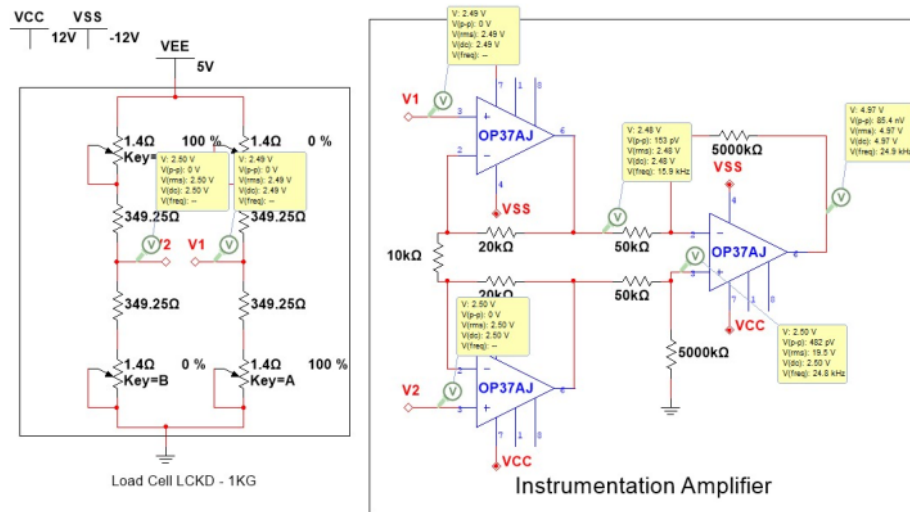


Figure 29 - 1KG Load Cell Amplifier

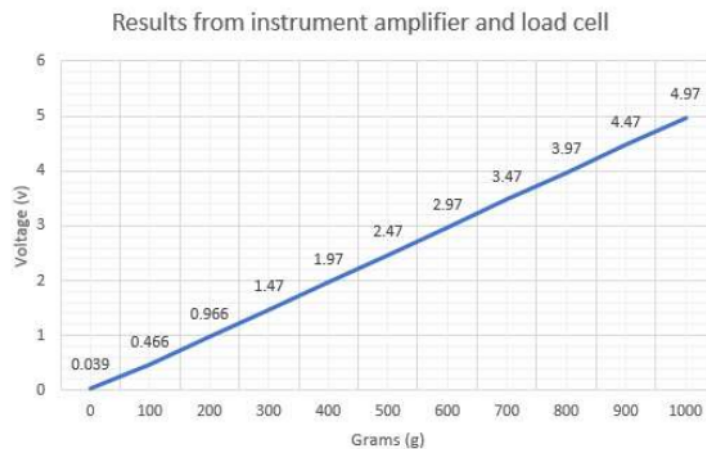
2.3 Testing.

The resistance began at 50% and is increased by 5% every time the weight is increased by 100 grams.

% Resistance	Grams (g)	Voltage (V)
50	0	0.039
55	100	0.466
60	200	0.966
65	300	1.47
70	400	1.97
75	500	2.47
80	600	2.97
85	700	3.47
90	800	3.97
95	900	4.47
100	1000	4.97

Good table of results

The table shows that when the resistance is increased the weight and voltage increase the voltage increases at a steady rate of around 0.5V. By looking at the results, the graph produced should be linear. *Grammatical error*



Good linear graph of the results

Figure 30 - Load Cell Graph

As predicted from the table, a linear graph was produced. As the grams increases the voltage also increases. In each 100g intervals the voltage is increased by almost 0.5V going to a maximum of 4.97V which is when the resistance is at 100%.

(See Appendix 4.2 for Tests)
6.2.1

There is good evidence of test results in the appendix

3. Prototypes

3.1 10 and 12-bit ADC prototype examples using LabVIEW.

Figure 31 shows A 10-bit ADC on LabVIEW. Two screens are shown, The panel view and the block diagram view. The analogue input is a slider which can vary from 0-5. Figure 30 shows [Figure 31](#) that the slider is halfway (2.5) giving a Boolean value of 511.5, the 10th LED is on which is 512. This proves that the ADC is working correctly, to back this another test was carried out when the analogue input is at max (5). Figure 32 shows the analogue input at 5, the Boolean value is 1023. All the LEDS are on as they all add up to the maximum value of 1023.



Figure 31 - 10-bit ADC (2.5)



Good full scale test

Figure 32 - 10-bit ADC (5)

The 10-bit ADC used in figure 30 was changed to a 12-bit ADC (2 LEDs added and the formula in the block diagram was changed)

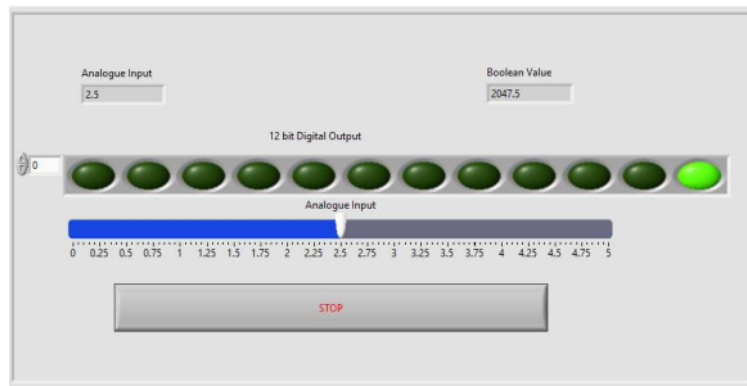
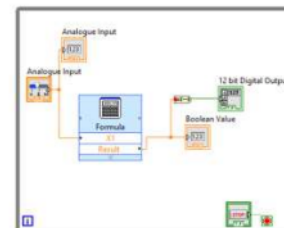


Figure 33 - 12-bit ADC (2.5)

Figure 33 shows the 12-bit ADC. The analogue input was set at half (2.5), since the maximum is 4096 a value of 2047.5 was achieved. Only the 12th LED is illuminating which is the 2048. This proves that its functioning correctly. Adding the two extra bits has increased the resolution to 4096 from 1024. The analogue input value was then set to 5.



Good full scale test

Figure 34 - 12-bit ADC (5)

Figure 34 shows the input set to 5 giving a Boolean value of 4095. All LEDs are illuminated as they all add up to a value of 4096 (maximum).

3.2 8-bit ADC with virtual analogue input

Figure 35 shows a panel view, block diagram and a potentiometer which can be used to manually adjust the analogue input. The panel view has 8 digital output LEDs, an analogue input which is controlled by the potentiometer. The results can then be viewed on the panel view, they're obtained from the formula placed in the formula block $(X1/5) * 256$.

X1 being the Analogue input signal.

5 being the maximum voltage.

256 being the 8-bit resolution.

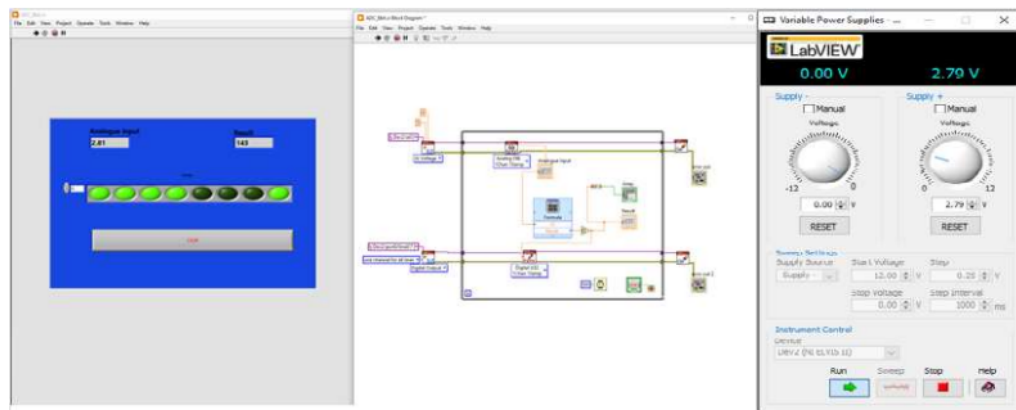


Figure 35 - 8-bit ADC

Good evidence of test setup

Figure 34 shows an analogue input of 2.81 which is giving a result value of 143 ($1+2+4+8+128$).

A similar test was carried out where the input was changed to max (See Figure 36). All LEDs are illuminated ($1+2+4+8+16+32+64+128 = 255$).

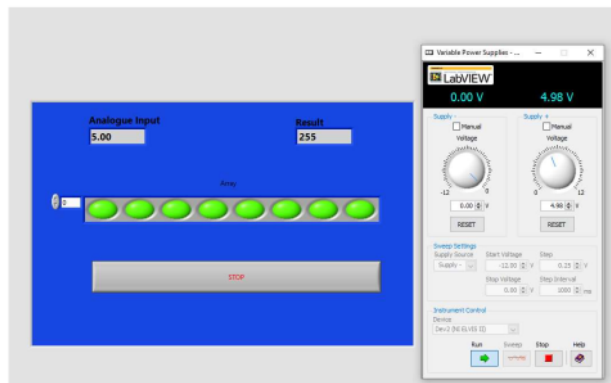


Figure 36 8-Bit ADC 2nd test

Good full scale test

4. Digital Magnitude Comparator

4.1. Nexys 4 DDR FPGA

The Nexys 4 DDR is a FPGA development board. The board is compatible with many development tools such as Vivado, Multisim, Xilinx and ISE. It can be programmed with VHDL or Verilog. The board is user friendly and has a range of onboard peripherals and connectors as well as programmable logic cells and I/O pins.

The FPGA will be used as a magnitude comparator with LabView also being used. The board will be connected to a NI Elvis II board, Figure 36 shows the pin arrangement.



Figure 20. Pmod connectors; front view, as loaded on PCB.

Table 5. Nexys4 DDR Pmod pin assignments.				
Pmod JA	Pmod JB	Pmod JC	Pmod JD	Pmod XDAC
JA1: C17	JB1: D14	JC1: K1	JD1: H4	JXADC1: A13 (AD3P)
JA2: D18	JB2: F16	JC2: F6	JD2: H1	JXADC2: A15 (AD10P)
JA3: E18	JB3: G16	JC3: J2	JD3: G1	JXADC3: B16 (AD2P)
JA4: G17	JB4: H14	JC4: G6	JD4: G3	JXADC4: B18 (AD11P)
JA7: D17	JB7: E16	JC7: E7	JD7: H2	JXADC7: A14 (AD3N)
JA8: E17	JB8: F13	JC8: J3	JD8: G4	JXADC8: A16 (AD10N)
JA9: F18	JB9: G13	JC9: J4	JD9: G2	JXADC9: B17 (AD2N)
JA10: G18	JB10: H16	JC10: E6	JD10: F3	JXADC10: A18 (AD11N)

Figure 37 – Pins

4.2 Magnitude Comparator setup

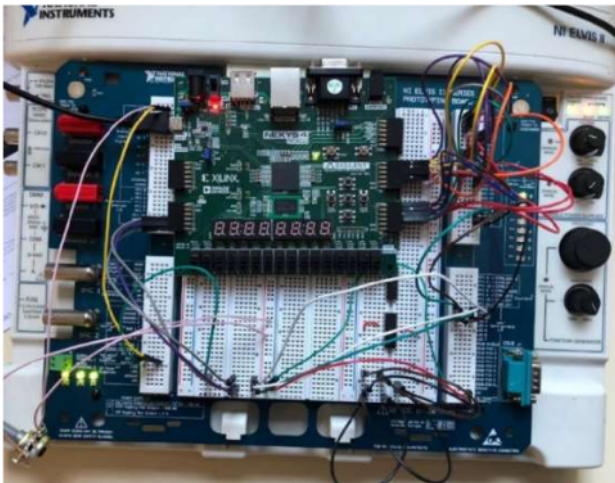


Figure 38 - FPGA connected to NI ELVIS

Good image of test setup

Interfacing ampping method is missing

Figure 38 shows the connection of the FPGA (magnitude comparator) and NI Elvis II. Links were placed from the FPGA ports to the Elvis to make it function correctly. The Elvis also contains a potentiometer (changes the magnitude comparator state).

This was simulated using the virtual PS

4.3 LabVIEW program

Figure 39 shows the block diagram of the LabVIEW program created for the magnitude comparator. The diagram contains many components, most are annotated. The diagram contains 4 formula blocks.

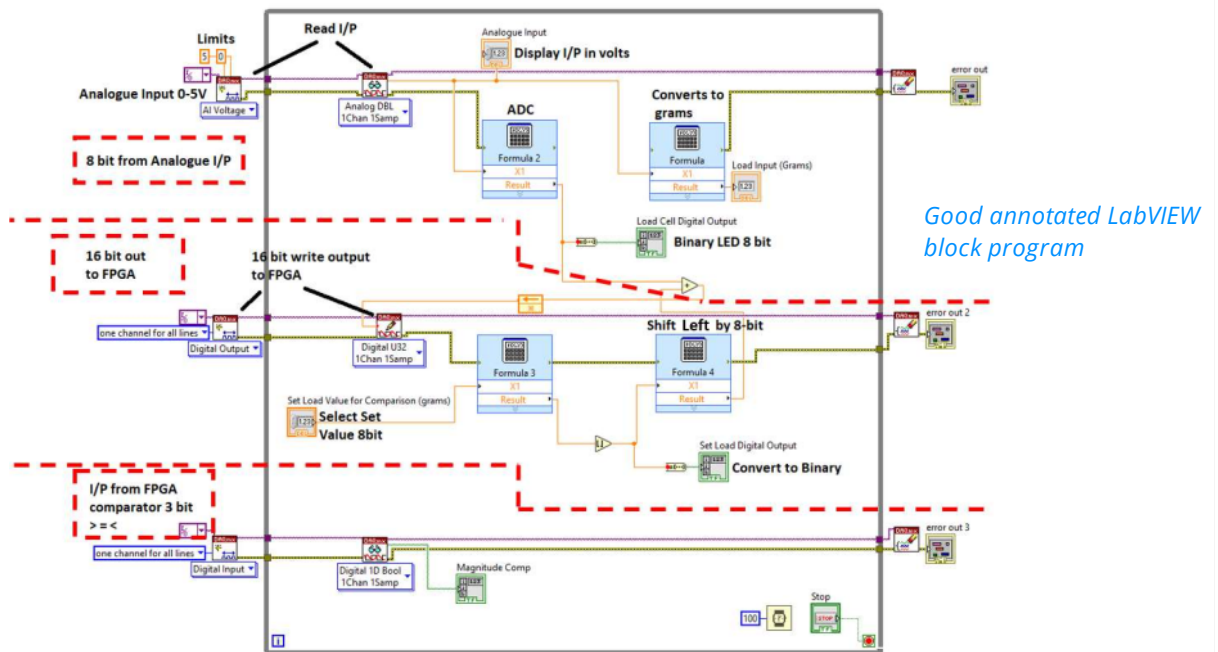


Figure 39 - Mag comb LabVIEW

The 8 bit shift left with example is not included

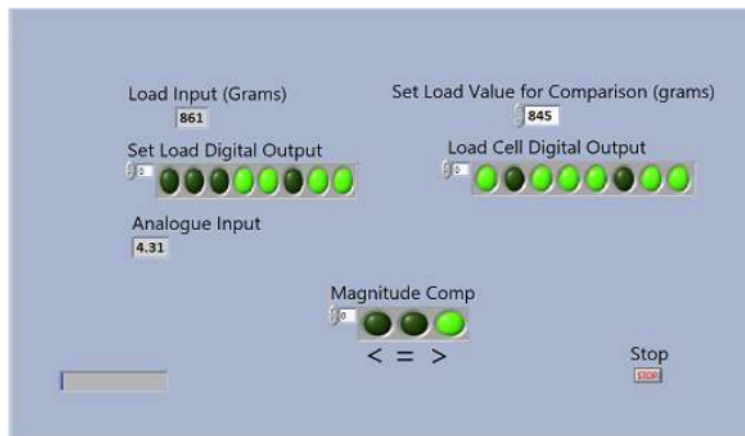
Formulas:

1. This is required to convert the analogue input into grams. This is achieved by dividing the analogue input with the maximum analogue input and multiplying the value by 1000.
2. This is required to convert the analogue input into a digital output. The analogue input is divided by the maximum analogue input and then multiplied by the resolution (256). The output is connected to a digital output which displays the gram value using the 8-bit word.

- This is required to convert the selected load value into a digital 8-bit word output. The set load value is divided by the maximum gram value (1Kg) then multiplied by the resolution. *The explanation for the formula blocks is incomplete*

4.4 Testing *The heading should be more specific*

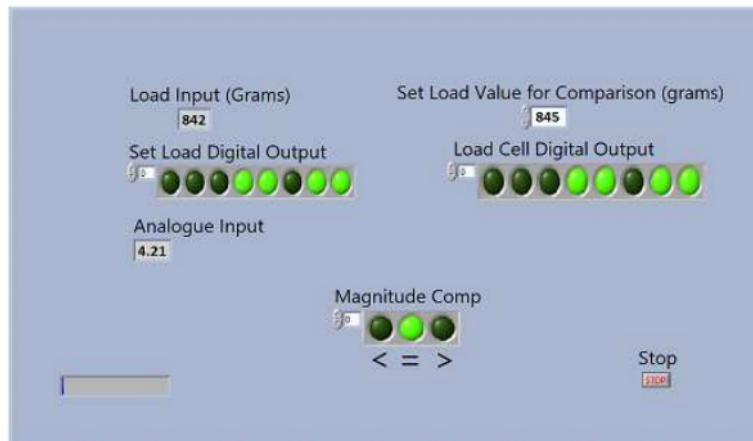
Greater Than



Test is successful and proves the correct result is input to the LabVIEW program from the FPGA.

Figure 40 - Greater Than

Equal To



Test is successful

Figure 41 - Equal to

Less Than

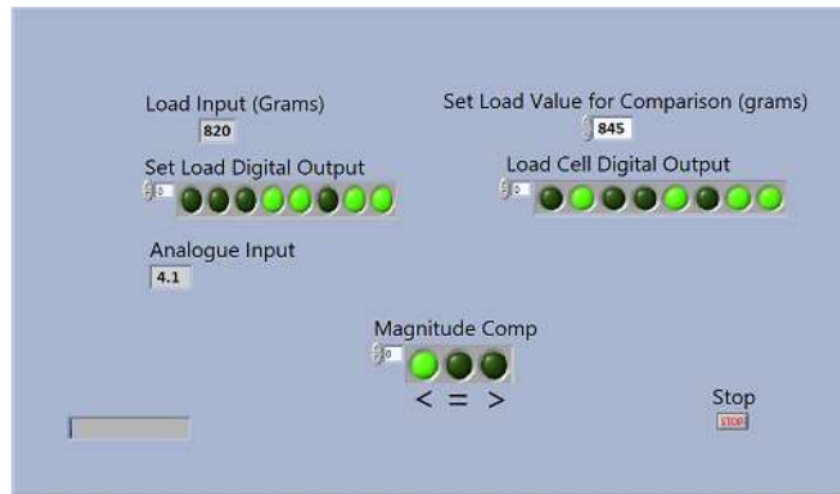


Figure 42 - Less Than

The 3 conditions have been tested successfully

Three tests were carried out, making sure the magnitude comparator is working correctly. The 'Set Load Value for Comparison' is used to manually input a value and the ADC converts this to the 'Load Input'. The set load was set to 845g. The three tests will be greater than, equal to and less than this value.

This is just for comparison. It is not the actual load value

Test 1 – Figure 40 shows that the 'Load Input' is 861g which is 16g heavier than the 845g set load. Due to the load input being higher the magnitude comparator has deemed this as greater than, which is correct.

Test 2 – Figure 41 shows that the 'Load Input' is 842g which is 3g lighter than the 845g set however the magnitude comparator has deemed this as equal as it is near to the set value, making the test successful.

Test 3 – Figure 42 shows that the 'Load Input' is 820g which is 25g lighter than the 845g set value. The magnitude comparator has deemed this as less than, which is correct.

Explanation here is good.

The evidence in this element indicates an understanding of the tasks carried out. The explanations in this element are improved. The headings should be more informative of the content.

5. Evaluation

Within this element the analogue and digital data's capabilities were heavily looked into. Analogue data was looked in to because of the load cell used, in this case the analogue data is represented by a continuous signal which is proportional to the weight being measured. The analogue data was conditioned for the ADC to convert it to binary. *This should be written using an improved method*

The resistance began at 50% and is increased by 5% every time the weight is increased by 100 grams. This produced a linear graph (see figure 30).

Comparisons were made between one digital value and another, this was achieved by using the magnitude comparator which determines whether the data is equal to, greater than or less than to a given digital value.

The LabVIEW testing of the magnitude comparator was successful. When altering the input of the load cell the magnitude comparator responds correctly to the changes, the new analogue input is compared with the set digital value given.

The evaluation is misleading, it lacks differentiation between tasks carried out.

Element 3

1. Introduction

The NI LabVIEW design used in Element 2 was modified to output the actual load cell values in grams over its full range. The comparison values were changed in to the form of data logging files. A graph was then produced on Excel to show the actual load over the range over a period. The purpose of this element is to make the program save the data over a specific time. The data is sent to a text file where the analogue input and output values can be logged.

Explanation here is poor

2. Design Requirements

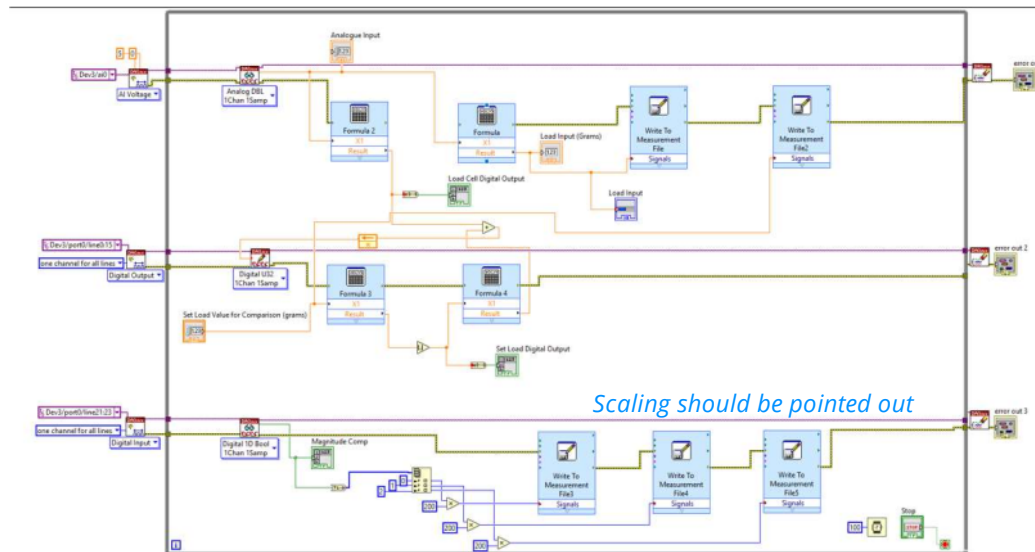


Figure 43 - Data Logging Block

Figure 43 shows the new program block. 'Write to measurement' blocks are connected to the data lines which will record and log to the text files. Four different blocks were added, below are their purposes.

Block 1 – This logs the timeline and load, spanning across the test period.

Block 2 – This logs the test value.

Block 3,4,5 – These blocks scale and log the comparators condition values. ($<$, $=$, $>$)

This is an improved explanation about the purpose of the 'write to file' blocks for data logging

Evidence of the file configurations are evident

Appendix 6.3.1 shows the 5 'Write to Measurement' file configurations. These are needed to record the data. File 1 is set to have a time column as it is logging the timeline.

4. Testing and Analysis

The setpoint was set to 900g. The comparator is comparing the analogue input of the potentiometer which is representing the load cell output.

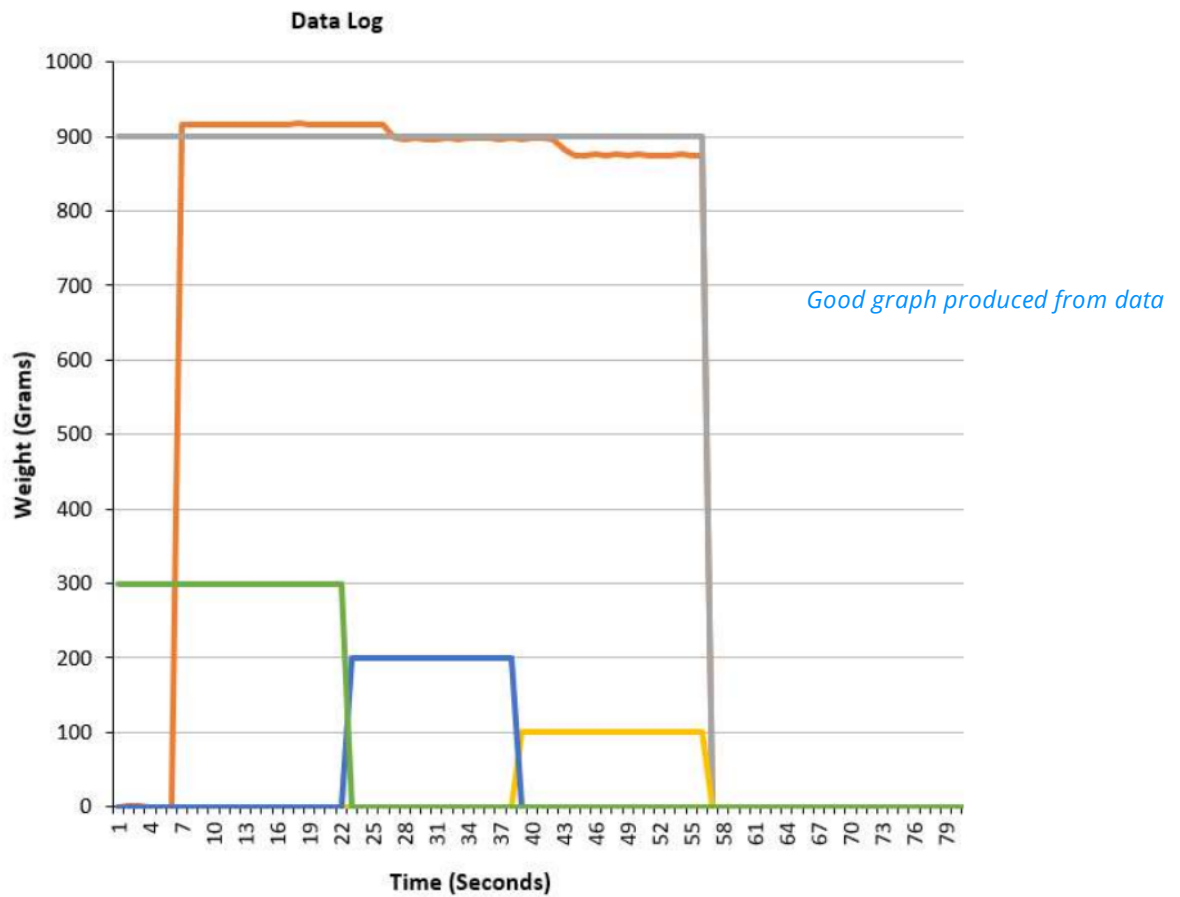


Figure 44 - Data Logging Graph

Figure 44 shows the graph produced from the data logging. The setpoint is 900g, the sensor data is higher than 900g up until 22 seconds. This can be seen by looking at the green trend which shows greater than. At 23 seconds the sensor data is equal to the setpoint and the blue trend is showing meaning that it is equal to. At around 40 seconds the sensor data is lower than the setpoint making the yellow trend appear.

There is no test data sample included. This is the data generated by the file write blocks, and needed for the graph creation.

5. Evaluation

The purpose of this element was to digitally compare the analogue input with the setpoint using the load cell. The LabVIEW program was implemented on to the Nexys FPGA board. A test was carried out testing a set load of 900g, this was successfully achieved based off the graph obtained from the logged data. The graph proves that the data from the program is logging correctly.

6. Conclusion and Recommendations

Research was carried out prior to developing the software, this assisted in gaining a better understanding of how the system works by researching in to components such as the digital magnitude comparator and load cells. Research on LabView was also carried out giving a better understanding of how to use it and knowing the purpose of important functions within the software.

A better understanding of how to use LabVIEW was achieved as a program was successfully created on it. It has also given a better understanding of how everything works through viewing the program visually and seeing the test results. Carrying out this experiment has allowed for the development of many skills such as programming, research skills, problem solving and critical thinking, skills which can be important in an industrial environment.

The portfolio has also shown the importance of prototypes as they can be used to test and validate design idea and concepts and assist in identifying potential issues early on in the design process.

A recommendation would be to use a higher resolution as on the testing of element 2, the set value was set to 845g, and the load input was 842g but it was reading it as equal to despite the 3g difference. This could be important in a system using an expensive or dangerous product.

Evaluation has relevant points and a good recommendation

Bibliography

Elprocus, 2022. *Analog to digital converter.* [Online]
Available at: <https://www.elprocus.com/analog-to-digital-converter/>
[Accessed 19th February 2023].

HBM, 2020. *What is a signal conditioner.* [Online]
Available at: <https://www.hbm.com/en/7339/what-is-a-signal-conditioner/>
[Accessed 20th February 2023].

MathWorks, 2022. *NI DAQmx.* [Online]
Available at: <https://uk.mathworks.com/hardware-support/nidaqmx.html>
[Accessed 19th February 2023].

Microcontrollerslab, 2022. *CD4063 4-bit magnitude comparator.* [Online]
Available at: <https://microcontrollerslab.com/cd4063-4-bit-comparator-pinout-examples-applications-datasheet/>
[Accessed 19th February 2023].

Nandland, 2018. *Tutorial - Introduction to VHDL.* [Online]
Available at: <https://nandland.com/introduction-to-vhdl-for-beginners-with-code-examples/>
[Accessed 18th February 2023].

NationInstruments, 2019. *LabVIEW Functions Overview.* [Online]
Available at: https://www.ni.com/docs/en-US/bundle/labview/page/lvconcepts/functions_overview.html
[Accessed 19th February 2023].

NationInstruments, 2022. *What is LabVIEW?.* [Online]
Available at: <https://www.ni.com/en-gb/shop/labview.html>
[Accessed 19th February 2023].

Omega, 2023. *LCKD.* [Online]
Available at: <https://www.omega.co.uk/pptst/LCKD.html>
[Accessed 19th February 2023].

Pandey, H., 2022. *Magnitude Comparator in Digital Logic.* [Online]
Available at: <https://www.geeksforgeeks.org/magnitude-comparator-in-digital-logic/>
[Accessed 19th February 2023].

Semiengineering, 2019. *Intellectual Property (IP)*. [Online]
Available at: https://semiengineering.com/knowledge_centers/intellectual-property/
[Accessed 18th February 2023].

An appropriate number of sources used

6.Appendix

6.1.1 4-bit code

```
library IEEE;
use IEEE.std_logic_1164.all;
entity adder4bit is
port(
cin : in STD_LOGIC;
a0 : in STD_LOGIC;
a1 : in STD_LOGIC;
a2 : in STD_LOGIC;
a3 : in STD_LOGIC;

b0 : in STD_LOGIC;
b1 : in STD_LOGIC;
b2 : in STD_LOGIC;
b3 : in STD_LOGIC;

cout : out STD_LOGIC;

s0 : out STD_LOGIC;
s1 : out STD_LOGIC;
s2 : out STD_LOGIC;
s3 : out STD_LOGIC
);
end adder4bit;

architecture adder4bit of adder4bit is

component fulladd
port (
a : in STD_LOGIC;
b : in STD_LOGIC;
cin : in STD_LOGIC;
cout : out STD_LOGIC;
s : out STD_LOGIC
);
end component;
```

```
signal c1 : STD_LOGIC;
signal c2 : STD_LOGIC;
signal c3 : STD_LOGIC;

begin

U1 : fulladd
port map(
a => a0, b => b0, cin => cin, cout => c1,
s => s0);
U2 : fulladd
port map(
a => a1, b => b1, cin => c1, cout => c2,
s => s1);
U3 : fulladd
port map(
a => a2, b => b2, cin => c2, cout => c3,
s => s2);
U4 : fulladd
port map(
a => a3, b => b3, cin => c3, cout => cout,
s => s3);

end adder4bit;
```

The code is not explained in the report

6.1.2 4-bit test parameters

```
a0 <= '1';
a1 <= '0';
a2 <= '0';
a3 <= '0';
b0 <= '1';
b1 <= '0';
b2 <= '0';
b3 <= '0';
cin <= '0';
wait for 10 ns;

a0 <= '1';
a1 <= '1';
a2 <= '1';
a3 <= '0';
b0 <= '0';
b1 <= '0';
b2 <= '1';
b3 <= '0';
cin <= '0';
wait for 10 ns;
```

Full range is not included here!

6.1.2 PLD VHDL export

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity mag4bit is
  port (
    B3 : in STD_LOGIC;
    B2 : in STD_LOGIC;
    B1 : in STD_LOGIC;
    B0 : in STD_LOGIC;
    A3 : in STD_LOGIC;
    A2 : in STD_LOGIC;
    A1 : in STD_LOGIC;
    A0 : in STD_LOGIC;
    AGTB : in STD_LOGIC;
    AEQB : in STD_LOGIC;
    ALTB : in STD_LOGIC;
    OAGTB : out STD_LOGIC;
    OAEQB : out STD_LOGIC;
    OALTB : out STD_LOGIC;
  );
end mag4bit;

architecture BEHAVIORAL of mag4bit is
begin
  PROCESS (A0, B0, A1, A2, A3, B1, B2, B3, AGTB, AEQB, ALTB)
  -----
  BEGIN
    if A3>B3 then OAGTB<='1';
      OAEQB<='0';
      OALTB<='0';
    elsif A3<B3 then OAGTB<='0';
      OAEQB<='0';
      OALTB<='1';
    elsif A3=B3 then
      if A2>B2 then OAGTB<='1';
        OAEQB<='0';
        OALTB<='0';
      elsif A2<B2 then OAGTB<='0';
        OAEQB<='0';
        OALTB<='1';
      elsif A2=B2 then
        if A1>B1 then OAGTB<='1';
          OAEQB<='0';
          OALTB<='0';
        elsif A1<B1 then OAGTB<='0';
          OAEQB<='0';
          OALTB<='1';
        elsif A1=B1 then
          if A0>B0 then OAGTB<='1';
            OAEQB<='0';
            OALTB<='0';
          elsif A0<B0 then OAGTB<='0';
            OAEQB<='0';
            OALTB<='1';
          else
            OAGTB<='1';
            OAEQB<='0';
            OALTB<='0';
          end if;
        end if;
      end if;
    end if;
  end if;
END PROCESS;
end BEHAVIORAL;

```

This code is not required as it is auto generated from PLD design in Msim.

6.1.3 Test Bench Export

```

entity tb_mag4bit is
end tb_mag4bit;

architecture tb of tb_mag4bit is

    component mag4bit
        port (S3 : in std_logic;
              S2 : in std_logic;
              S1 : in std_logic;
              S0 : in std_logic;
              A3 : in std_logic;
              A2 : in std_logic;
              A1 : in std_logic;
              A0 : in std_logic;
              AGTB : in std_logic;
              AEQB : in std_logic;
              ALTB : in std_logic;
              OAGTB : out std_logic;
              OAEQB : out std_logic;
              OALTb : out std_logic);
    end component;

    signal S3 : std_logic;
    signal S2 : std_logic;
    signal S1 : std_logic;
    signal S0 : std_logic;
    signal A3 : std_logic;
    signal A2 : std_logic;
    signal A1 : std_logic;
    signal A0 : std_logic;
    signal AGTB : std_logic;
    signal AEQB : std_logic;
    signal ALTB : std_logic;
    signal OAGTB : std_logic;
    signal OAEQB : std_logic;
    signal OALTb : std_logic;

begin
    dut : mag4bit
        port map (S3 => S3,
                  S2 => S2,
                  S1 => S1,
                  S0 => S0,
                  A3 => A3,
                  A2 => A2,
                  A1 => A1,
                  A0 => A0,
                  AGTB => AGTB,
                  AEQB => AEQB,
                  ALTB => ALTB,
                  OAGTB => OAGTB,
                  OAEQB => OAEQB,
                  OALTb => OALTb);

    stimuli : process
    begin
        -- EDIT Adapt Initialization as needed
        S3 <= '1';
        S2 <= '1';
        S1 <= '0';
        S0 <= '0';
        A3 <= '1';
        A2 <= '1';
        A1 <= '0';
        A0 <= '0';
        AGTB <= '0';
        AEQB <= '1';
        ALTB <= '0';
        wait for 100ns;
        S3 <= '1';
        S2 <= '1';
        S1 <= '0';
        S0 <= '1';
        A3 <= '1';
        A2 <= '1';
        A1 <= '0';
        A0 <= '1';
        AGTB <= '0';
        AEQB <= '1';
        ALTB <= '0';
        wait for 100ns;
        S3 <= '1';
        S2 <= '0';
        S1 <= '0';
        S0 <= '1';
        A3 <= '1';
        A2 <= '0';
        A1 <= '0';
        A0 <= '0';
        AGTB <= '0';
        AEQB <= '1';
        ALTB <= '0';
        wait for 100ns;
        -- EDIT Add stimuli here
        S3 <= '0';
        S2 <= '0';
        S1 <= '1';
        S0 <= '0';
        A3 <= '0';
        A2 <= '0';
        A1 <= '1';
        A0 <= '0';
        AGTB <= '0';
        AEQB <= '1';
        ALTB <= '0';
        wait for 100ns;
        end process;
end tb;

```

6.1.4 Mag4bit constraints extract

```

##Switches
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { A0 }]; #IO_L24N_T3_R50_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { A1 }]; #IO_L34L_T0_D05_PPCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { A2 }]; #IO_L64L_T0_D06_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { A3 }]; #IO_L13N_T2_M0CC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN T17 IOSTANDARD LVCMOS33 } [get_ports { S0 }]; #IO_L12N_T2_M0CC_14 Sch=sw[4]
set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { S1 }]; #IO_L7N_T1_D10_14 Sch=sw[5]
set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { S2 }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { S3 }]; #IO_L54L_T0_D07_14 Sch=sw[7]
set_property -dict { PACKAGE_PIN T8 IOSTANDARD LVCMOS18 } [get_ports { c0 }]; #IO_L24N_T3_34 Sch=sw[8]
set_property -dict { PACKAGE_PIN U8 IOSTANDARD LVCMOS18 } [get_ports { c1 }]; #IO_25_34 Sch=sw[9]
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { c2 }]; #IO_L15P_T2_D05_R0MR_B_14 Sch=sw[10]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { c3 }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCMOS33 } [get_ports { S02 }]; #IO_L24P_T3_35 Sch=sw[12]
set_property -dict { PACKAGE_PIN U12 IOSTANDARD LVCMOS33 } [get_ports { AGTB }]; #IO_L20P_T2_A08_D24_14 Sch=sw[13]
set_property -dict { PACKAGE_PIN U11 IOSTANDARD LVCMOS33 } [get_ports { AEQB }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { cin }]; #IO_L21P_T3_D05_14 Sch=sw[15]

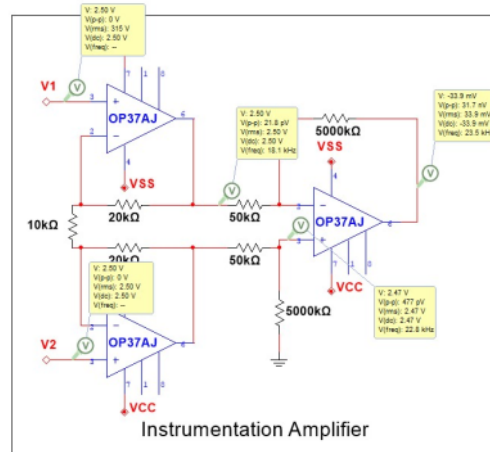
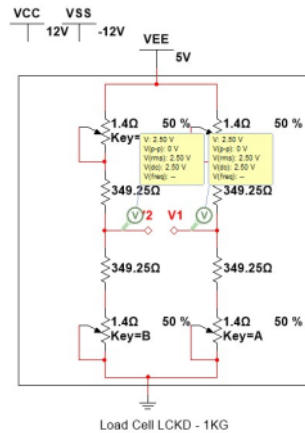
## LEDs
set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { OAGTB }]; #IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { OAEQB }]; #IO_L24P_T3_R51_15 Sch=led[1]
set_property -dict { PACKAGE_PIN T13 IOSTANDARD LVCMOS33 } [get_ports { OALTb }]; #IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN R14 IOSTANDARD LVCMOS33 } [get_ports { led3 }]; #IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { led4 }]; #IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { led5 }]; #IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { led6 }]; #IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { led7 }]; #IO_L18P_T2_A12_D28_14 Sch=led[7]
set_property -dict { PACKAGE_PIN V16 IOSTANDARD LVCMOS33 } [get_ports { LED8 }]; #IO_L16N_T2_A15_D31_14 Sch=led[8]
set_property -dict { PACKAGE_PIN T15 IOSTANDARD LVCMOS33 } [get_ports { LED9 }]; #IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { LED10 }]; #IO_L22P_T3_A05_D21_14 Sch=led[10]
set_property -dict { PACKAGE_PIN T16 IOSTANDARD LVCMOS33 } [get_ports { LED11 }]; #IO_L15N_T2_D05_D0UT_C50_9_14 Sch=led[11]
set_property -dict { PACKAGE_PIN V15 IOSTANDARD LVCMOS33 } [get_ports { LED12 }]; #IO_L16P_T2_C51_B_14 Sch=led[12]
set_property -dict { PACKAGE_PIN V14 IOSTANDARD LVCMOS33 } [get_ports { LED13 }]; #IO_L22N_T3_A04_D20_14 Sch=led[13]

```

Constarints file sample is not explained or ref from report

6.2.1 – Load Cell testing

50%

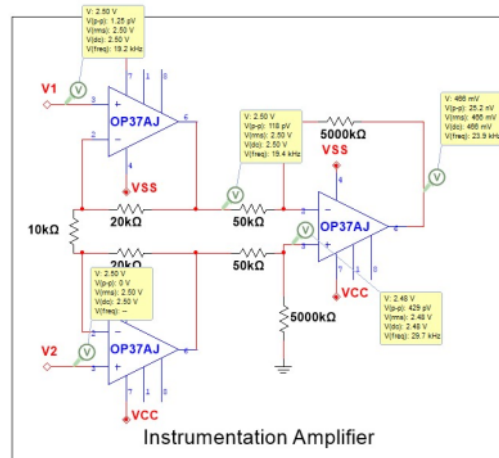
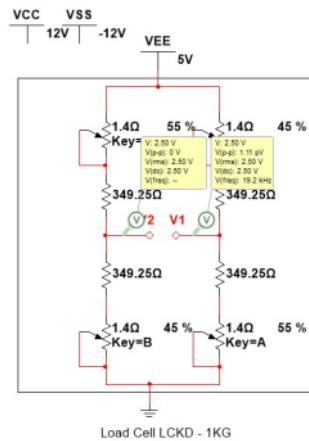


Good evidence

V: -33.9 mV
V(p-p): 31.7 nV
V(rms): 34.0 mV
V(dc): -33.9 mV
V(freq): 23.5 kHz

Vrms = 33.9mV (0V) for equal load

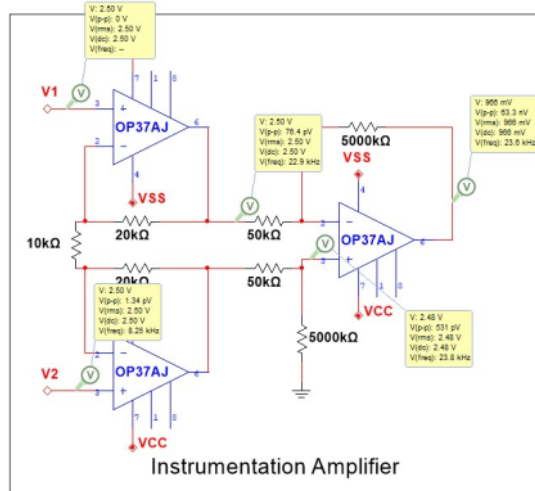
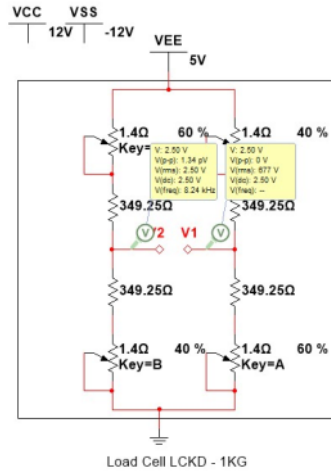
55%



V: 466 mV
V(p-p): 56.8 nV
V(rms): 466 mV
V(dc): 466 mV
V(freq): 23.9 kHz

Vrms = 466mV for load of 55% to 45%

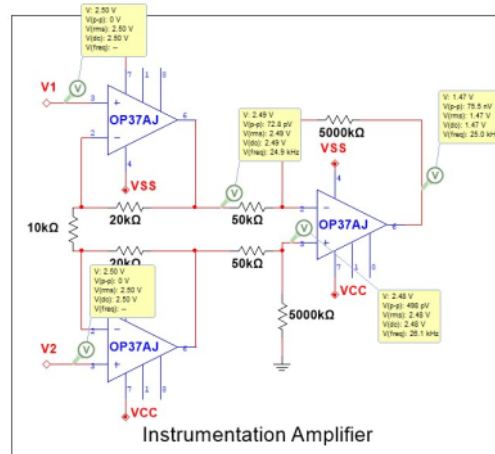
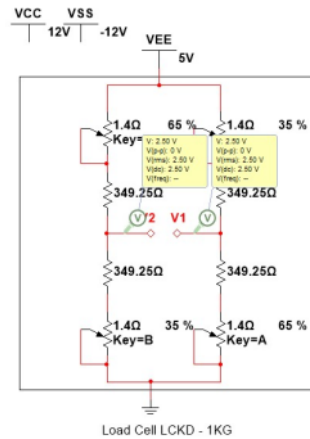
60%



V: 966 mV
V(p-p): 28.5 nV
V(rms): 1.01 V
V(dc): 966 mV
V(freq): 23.1 kHz

Vrms = 966mV for load of 60% to 40%

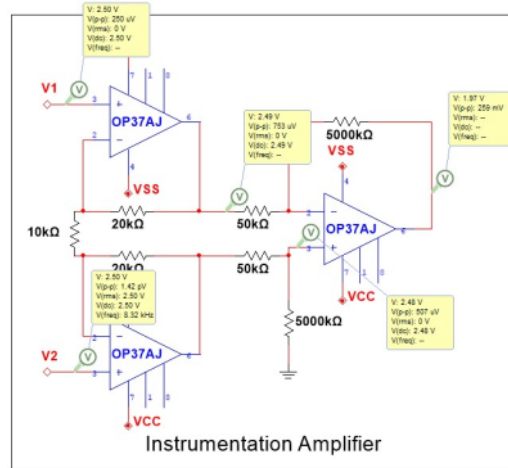
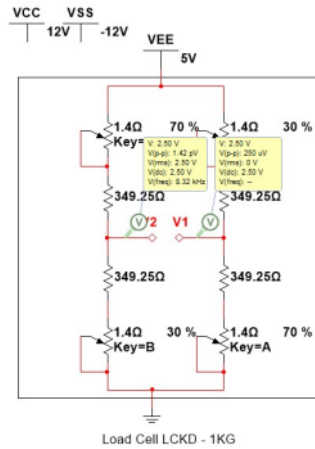
65%



V: 1.47 V
V(p-p): 66.5 nV
V(rms): 1.47 V
V(dc): 1.47 V
V(freq): 25.1 kHz

Vrms = 1.47V for load of 65% to 35%

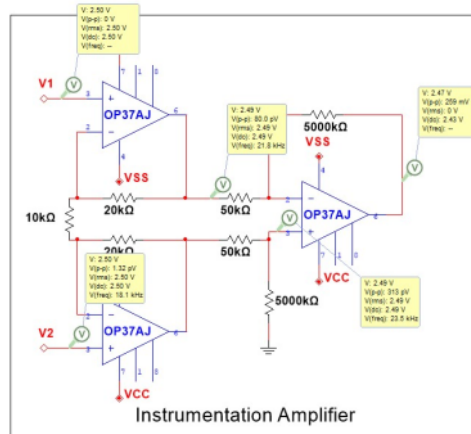
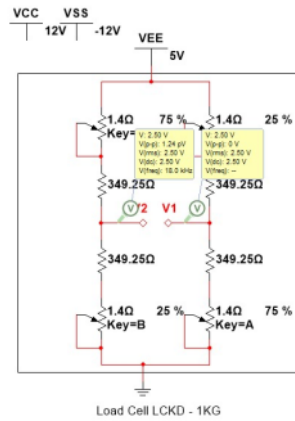
70%



V: 1.97 V
V(p-p): 42.4 nV
V(rms): 1.97 V
V(dc): 1.97 V
V(freq): 25.1 kHz

Vrms = 1.97V for load of 70% to 30%

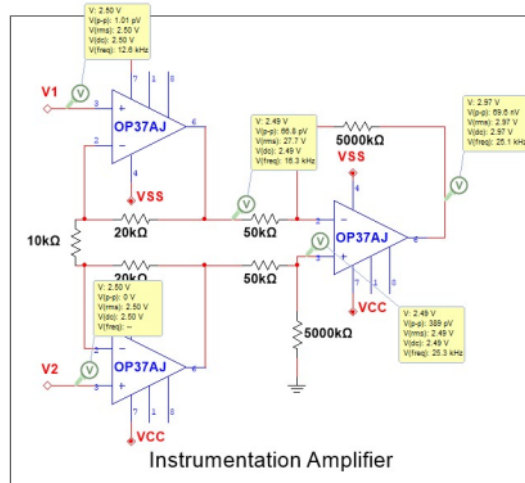
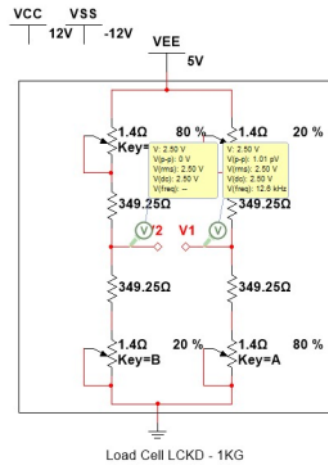
75%



V: 2.47 V
V(p-p): 52.8 nV
V(rms): 2.47 V
V(dc): 2.47 V
V(freq): 25.7 kHz

Vrms = 2.47V for load of 75% to 25%.

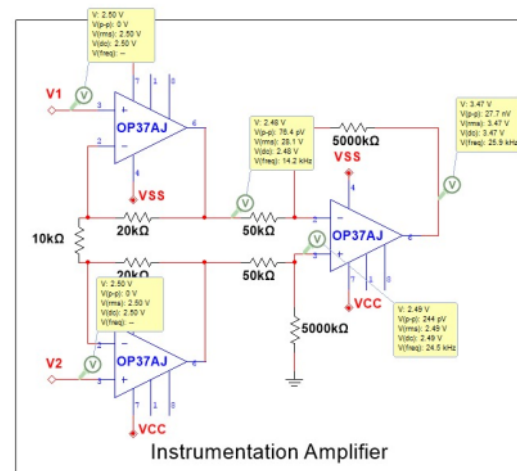
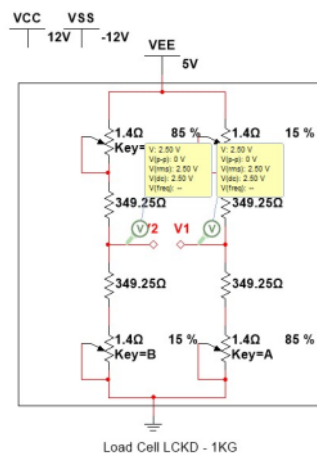
80%



V: 2.97 V
V(p-p): 45.3 nV
V(rms): 2.97 V
V(dc): 2.97 V
V(freq): 25.3 kHz

Vrms = 2.97V for load of 80% to 20%.

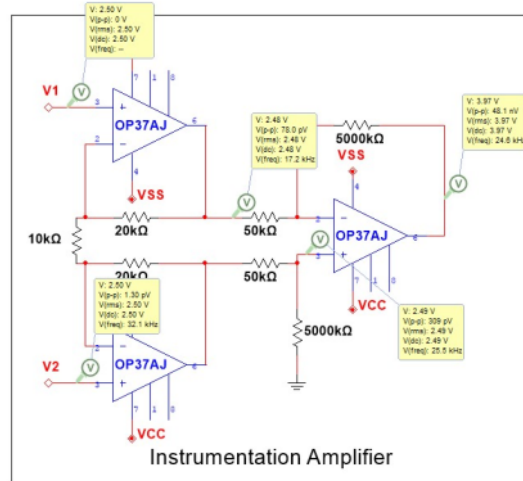
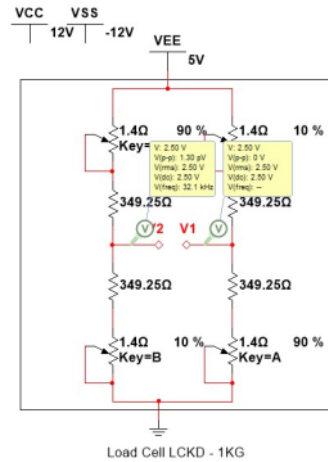
85%



V: 3.47 V
V(p-p): 56.8 nV
V(rms): 3.47 V
V(dc): 3.47 V
V(freq): 25.9 kHz

Vrms = 3.47V for load of 85% to 15%.

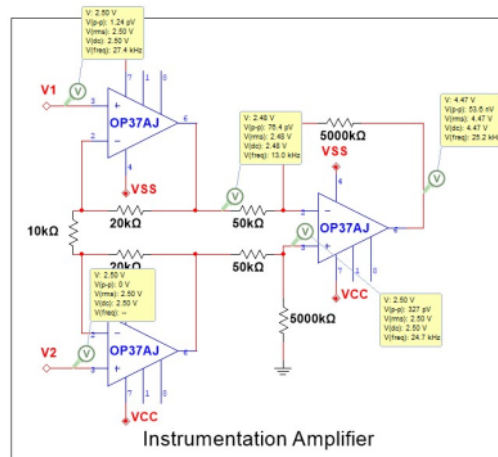
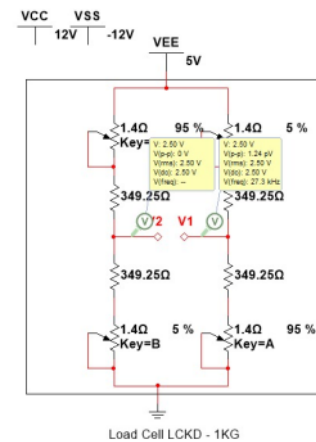
90%



V: 3.97 V
V(p-p): 41.3 nV
V(rms): 5.97 V
V(dc): 3.97 V
V(freq): 23.9 kHz

Vrms = 5.97V for load of 90% to 10%

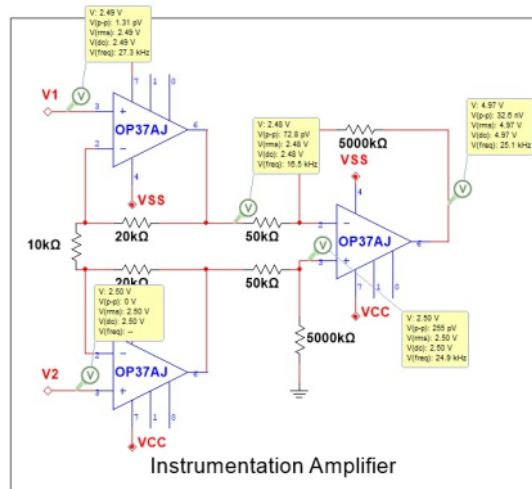
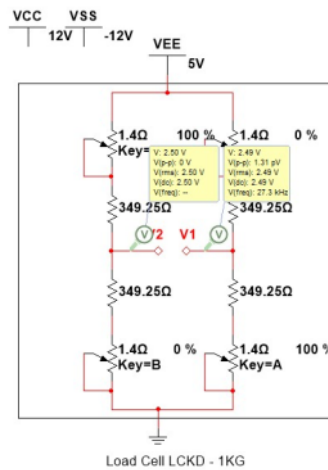
95%



V: 4.47 V
V(p-p): 89.6 nV
V(rms): 4.47 V
V(dc): 4.47 V
V(freq): 25.6 kHz

Vrms = 4.47V for load of 95% to 5%

100%



V: 4.97 V
V(p-p): 63.9 nV
V(rms): 4.97 V
V(dc): 4.97 V
V(freq): 25.4 kHz

Vrms = 4.97V for load of 100% to 0%

Good evidence of test results

6.3.1

File 1

Configure Write To Measurement File [Write To Measurement File]

Filename
H:\LabVIEW Data\test1.lvm

Action
☒ Save to one file
☐ Ask user to choose file
☒ Ask only once
☐ Ask each iteration
If a file already exists
☒ Rename existing file
☐ Use next available filename
☐ Append to file
☐ Overwrite file
☐ Save to series of files (multiple files)
Settings...

File Format
☒ Text (LVM)
☐ Binary (TDMS)
☐ Binary with XML Header (TDM)
☐ Microsoft Excel (xlsx)
☒ Lock file for faster access

Segment Headers
☐ One header per segment
☒ One header only
☐ No headers

X Value (Time) Columns
☐ One column per channel
☒ One column only
☐ Empty time column

Delimiter
☒ Tabulator
☐ Comma

File Description
Advanced...

OK Cancel Help

File 2

Configure Write To Measurement File [Write To Measurement File2]

Filename
H:\LabVIEW Data\test2.lvm

Action
☒ Save to one file
☐ Ask user to choose file
☒ Ask only once
☐ Ask each iteration
If a file already exists
☒ Rename existing file
☐ Use next available filename
☐ Append to file
☐ Overwrite file
☐ Save to series of files (multiple files)
Settings...

File Format
☒ Text (LVM)
☐ Binary (TDMS)
☐ Binary with XML Header (TDM)
☐ Microsoft Excel (xlsx)
☒ Lock file for faster access

Segment Headers
☐ One header per segment
☒ One header only
☐ No headers

X Value (Time) Columns
☐ One column per channel
☐ One column only
☒ Empty time column

Delimiter
☒ Tabulator
☐ Comma

File Description
Advanced...

OK Cancel Help

File 3

Configure Write To Measurement File [Write To Measurement File3]

Filename
H:\LabVIEW Data\test3.lvm

Action
☒ Save to one file
☐ Ask user to choose file
☒ Ask only once
☐ Ask each iteration
If a file already exists
☒ Rename existing file
☐ Use next available filename
☐ Append to file
☐ Overwrite file
☐ Save to series of files (multiple files)
Settings...

File Format
☒ Text (LVM)
☐ Binary (TDMS)
☐ Binary with XML Header (TDM)
☐ Microsoft Excel (.xlsx)
☒ Lock file for faster access

Segment Headers
☐ One header per segment
☒ One header only
☐ No headers

X Value (Time) Columns
☐ One column per channel
☐ One column only
☒ Empty time column

Delimiter
☒ Tabulator
☐ Comma

File Description
Advanced...

OK Cancel Help

File 4

Configure Write To Measurement File [Write To Measurement File4]

Filename
H:\LabVIEW Data\test4.lvm

Action
☒ Save to one file
☐ Ask user to choose file
☒ Ask only once
☐ Ask each iteration
If a file already exists
☒ Rename existing file
☐ Use next available filename
☐ Append to file
☐ Overwrite file
☐ Save to series of files (multiple files)
Settings...

File Format
☒ Text (LVM)
☐ Binary (TDMS)
☐ Binary with XML Header (TDM)
☐ Microsoft Excel (.xlsx)
☒ Lock file for faster access

Segment Headers
☐ One header per segment
☒ One header only
☐ No headers

X Value (Time) Columns
☐ One column per channel
☐ One column only
☒ Empty time column

Delimiter
☒ Tabulator
☐ Comma

File Description
Advanced...

OK Cancel Help

File 5

 Configure Write To Measurement File [Write To Measurement File5]

Filename

H:\LabVIEW Data\test5.lvm 

Action

☒ Save to one file

☐ Ask user to choose file

☒ Ask only once

☐ Ask each iteration

If a file already exists

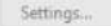
☒ Rename existing file

☐ Use next available filename

☐ Append to file

☐ Overwrite file

☐ Save to series of files (multiple files)



File Format

☒ Text (LVM)

☐ Binary (TDMS)

☐ Binary with XML Header (TDM)

☐ Microsoft Excel (xlsx)

☒ Lock file for faster access

Segment Headers

☐ One header per segment

☒ One header only

☐ No headers

X Value (Time) Columns

☐ One column per channel

☐ One column only

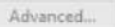
☒ Empty time column

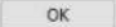
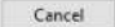
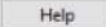
Delimiter

☒ Tabulator

☐ Comma

File Description



FINAL GRADE

GENERAL COMMENTS

Instructor

51 /100

PAGE 1

PAGE 2

PAGE 3

Text Comment. A comprehensive table of contents

PAGE 4

PAGE 5

Text Comment. Extensive list of figures

PAGE 6

Text Comment. The 4 bit adder should be included in the introduction. You must be aware of the separate tasks required.

PAGE 7

Text Comment. Good significant research carried out here with relevant sources, and good example of IP system.

PAGE 8

Text Comment. DAQ program blocks have been explained and are significant for the practical in this portfolio. There are aother blocks that should have been include such as formula blocks.

Text Comment. Significant points are explained using a relevant supporting source and example

Text Comment. The ADC methods such as pipeline, flash and SAR should be included

Text Comment. This should include a supporting image

Text Comment. A good understanding demonstrated with relevant source. A supporting image should be included

Text Comment. The point is made about the building blocks for the 4 bit adder

Text Comment. The block diagram is correct and is significant for the VHDL design.

Text Comment. This hierarchical listing here should be explained

Text Comment. There is no point in these images without an explanation

Text Comment. The significance of this 4 bit comparator simulation should be pointed out

Text Comment. The IP design is correct, however the explanation is poor

Text Comment. You must explain the purpose for the 8 bit mag comparator

Text Comment. This does not show the hierarchy

Text Comment. This combination IP design is correct. However, You have failed to explain the 4 bit adder and its function

PAGE 18

Text Comment. 8 bit PLD is correct and tested

Text Comment. There should be an explanation here!

PAGE 19

Text Comment. Full range is not tested. The carry bit 'cout' is not set

Text Comment. Appendix 6.1.3

Text Comment. Full range has been tested

PAGE 20

Text Comment. This needs to be explained further and include all 3 output conditions

Text Comment. The test is successful and the mapping indicates the io conditions

Text Comment. Correct result with io conditions set

PAGE 21

Text Comment. Test is successful with io conditions set

Text Comment. Good use of table for results

Text Comment. Explanations should be included

PAGE 22

Text Comment. This explanation is incomplete

Text Comment. Test is successful and io conditions set

Text Comment. Test is succesful with io conditions set

PAGE 23

Text Comment. Test is successful with io conditions set

Text Comment. Good use of table for results

Text Comment. There should be an explanation of the tests carried out

PAGE 24

Text Comment. Evaluation includes significant points

PAGE 25

Text Comment. element

Text Comment. Good introduction that demonstrates an understanding of task requirements

PAGE 26

Text Comment. Figure 28

Text Comment. Good explanation

Text Comment. Good test for unity gain

PAGE 27

Text Comment. Load cell spec is identified

Text Comment. Good choice of resistor values for instrument amplifier. The gain calc works out correctly and meets requirements

Text Comment. This should include that these are the initial conditions

PAGE 28

PAGE 29

Text Comment. Good table of results

Text Comment. Grammatical error

Text Comment. Good linear graph of the results

Text Comment. There is good evidence of test results in the appendix

Text Comment. 6.2.1

PAGE 30

Text Comment. Figure 31

Text Comment. Good full scale test

PAGE 31

Text Comment. Good full scale test

PAGE 32

Text Comment. Good evidence of test setup

Text Comment. Good full scale test

PAGE 33

Text Comment. Good image of test setup

Text Comment. Interfacing ampping method is missing

PAGE 34

Text Comment. This was simulated using the virtual PS

Text Comment. Good annotated LabVIEW block program

Text Comment. The 8 bit shift left with example is not included

PAGE 35

Text Comment. The explanation for the formula blocks is incomplete

Text Comment. The heading should be more specific

Text Comment. Test is successful and proves the correct result is input to the LabVIEW program from the FPGA.

Text Comment. Test is successful

PAGE 36

Text Comment. The 3 conditions have been tested successfully

Text Comment. This is just for comparison. It is not the actual load value

Text Comment. Explanation here is good.

Text Comment. The evidence in this element indicates an understanding of the tasks carried out. The explanations in this element are improved. The headings should be more informative of the content.

PAGE 37

Text Comment. This should be written using an improved method

Text Comment. The evaluation is misleading, it lacks differentiation between tasks carried out.

PAGE 38

Text Comment. Explanantion here is poor

Text Comment. Scaling should be pointed out

Text Comment. This is an improved explanation about the purpose of the 'wirte to file' blocks for data logging

PAGE 39

Text Comment. Evidence of the file configurations are evident

PAGE 40

Text Comment. Good graph produced from data

Text Comment. There is no test data sample included. This is the data generated by the fil wrtie blocks, and neede for the graph creation.

PAGE 41

PAGE 42

Text Comment. Evaluation has relevant points and a good recommendation

PAGE 43

PAGE 44

Text Comment. An appropriate number of sources used

PAGE 45

Text Comment. The code is not explained in the report

Text Comment. Full range is not included here!

PAGE 46

Text Comment. This code is not required as it is auto generated from PLD design in Msim.

PAGE 47

Text Comment. Constarints file sample is not explained or ref from report

PAGE 48

Text Comment. Good evidence

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

Text Comment. Good evidence of test results

PAGE 54

PAGE 55

PAGE 56

HARRI JONES**0****RESEARCH-INFORMED LITERATURE**

Research includes significant points with ref sources. Limited content for DAQ, ADC and Load cell.

5/10**KNOWLEDGE AND UNDERSTANDING**

Element 1 includes significant parts of the requirements. However there are images with no explanations, disjointed content and the 4 bit adder explanation is missing. The explanations are very brief and poor. 4/10 Element 2 instrument amplifier has correct resistor values chosen. Prototypes are all present and tested. LabVIEW program explanation is incomplete. 6/10 Element 3 explanations include significant points in a logical order. Some test data evidence is missing, however, the graph is produced. 5/10

15/30**ANALYSIS**

Element 1 test results have not been fully achieved. The 4 bit adder has not been tested for the carry over bit. The test results have not been explained

13/25

fully. The evaluation includes relevant points. 2.5/5
Element 2 test results are complete and include the required range of values. Evaluation is poor.
5.5/10 Element 3 test results are good and a graph is produced from test data. The evidence of test data is missing and not explained, there should be a sample at least. Evaluation has relevant points.
5/10

PRACTICAL APPLICATION AND DEP.



Element 1 design and build has been carried out. The explanations are poor for this section and content is disjointed. The 4 bit adder is not explained even though the program listing is evident in the appendix. The explanations should be far better in this element. 2/5 Element 2 interface mapping method is missing. However the task has been carried out. The program design is good and annotated. The explanation of the 8 bit shift left method is missing. 2.5/5 Element 3 data log method has been implemented with effective scaler values used. Test data generated from text files is not included as evidence. 5.5/10

10/20

SKILLS FOR PROFESSIONAL PRAC.



Element 1 has disjointed content and misleading headings. There are parts missing and grammatical errors. 2.5/5 Element 2 is an improvement and explanations are more relevant. there is a sequence of tasks carried out. 3/5 Element 3 has the required results in a sequence. There are parts missing in this section. 2.5/5

8/15

